

Version 4.01

Microsoft[®] MS-DOS[®]

**User's Guide &
User's Reference**

Microsoft® MS-DOS®

User's Guide

**Operating System
Version 4.01**

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software described in this document is furnished under a license agreement or non-disclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or non-disclosure agreement. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose without the express written permission of Microsoft.

©Copyright Microsoft Corporation, 1987, 1988. All rights reserved.

Simultaneously published in the United States and Canada.

Microsoft, the Microsoft logo, MS-DOS, XENIX, Multiplan, and GW-BASIC are registered trademarks of Microsoft Corporation.

Lotus is a registered trademark of Lotus Development Corporation.

MS-DOS User's Guide

Table of Contents

Welcome	v
Before You Begin	vi
Notational Conventions	vii
How to Use This Guide	vii
1 Learning About MS-DOS	1
Terms You Should Know	2
Program	2
File	2
Filename	3
Directory	4
Volume Label	4
Disk Drive	4
Drive Name	5
The Default Drive and the MS-DOS Prompt	5
Command	6
Devices	6
Device Names	6
Error Messages	7
Memory	7
Keys You Use with MS-DOS	8
The Enter Key	8
Keys that Move the Cursor	8
Control Key Combinations	9
2 Learning About Disks, Files, and Directories	11
Floppy Disks	12
Disk Protection	13
3.5inch Disks	14
Hard Disks	15
The Format Command	16
How to Name Your Files	16
Invalid Filenames	17
Directories	17

iv Contents

3	Getting Started 19
	How to Install MS-DOS Using the Select Program 19
	Running the Select Program 22
	The Function and Workspace Screen 22
	The Country and Keyboard Screen 22
	The Select Installation Drive Screen 22
	The Number of Printers Screen 23
	The MS-DOS Shell Option Screen 23
	The Installation Options Screen 23
	Continuing MS-DOS Installation 24
	An Overview of the MS-DOS Shell 25
	How to Start the MS-DOS Shell 25
	How to Quit MS-DOS 26
	How to Make a Backup Copy of Your MS-DOS Disk 26
4	Using Commands 29
	File Commands 29
	The Dir Command 29
	The Copy Command 31
	The Delete Command 32
	The Rename Command 33
	The Type Command 33
	The Print Command 35
	Disk Commands 36
	The Format Command 36
	The Diskcopy Command 38
5	Using Applications with MS-DOS 41
	How to Run Application Programs 41
	A Note About Using Application Programs 43
	How to Create a File with Edlin 43
6	Setting Up MS-DOS 47
	The Config.sys File 47
	The Autoexec.bat File 49
	How These Special Files Differ 50
	Summary 51
	Terms 53
	Index 59

Welcome

Welcome to the MS-DOS® operating system, version 4.0. If you are new at working with operating systems for personal computers, you will want to learn a few basics before you go on to learn about the advanced features of MS-DOS. This *MS-DOS® User's Guide* was written to help you understand the fundamentals of using MS-DOS with your personal computer.

Once you have read this *MS-DOS® User's Guide*, or if you are already an advanced personal computer user, see the *MS-DOS® User's Reference*, which presents the features of MS-DOS, and describes each MS-DOS command in detail. In addition, programmers may be interested in the *MS-DOS® Programmer's Reference*, which introduces programmers to MS-DOS, and describes all the MS-DOS system calls.

Also see these manuals

Before You Begin

System requirements

Before you begin using this *MS-DOS User's Guide*, you should have

- A 16-bit personal computer that runs MS-DOS and has at least 256K bytes of memory. Some 8-bit and 32-bit computers also support MS-DOS 4.0.
- One or two disk drives (either one or two floppy disk drives or one hard disk drive and one floppy disk drive).
- The *MS-DOS User's Reference*
- The MS-DOS master disks.

The following are the installation disk titles included in each set of master disks. They vary depending on the size of the disks used by your system.

5.25-inch disks (360 kilobytes)	3.5-inch disks (720 kilobytes)
Install (write protected)	Install (write protected)
Select	Operating
Operating1	MS-DOS Shell
Operating2	
Operating3	
MS-DOS Shell	

Notational Conventions

Throughout this manual, the following conventions are used to distinguish elements of text:

About this manual

bold	Used for commands, options, switches, and literal portions of syntax that must appear exactly as shown
<i>italic</i>	Used for filenames, variables, and placeholders that represent the type of text to be entered by the user
monospace	Used for sample command lines, program code and examples, and sample sessions
SMALL CAPS	Used for keys, key sequences, and acronyms

How to Use This Guide

This manual introduces you to the MS-DOS operating system and teaches you how to use several MS-DOS features. The manual is organized so that you can easily find what you need to know, as in the following list, which gives you a quick overview of the topics covered.

Turn to	To learn
Chapter 1	About your keyboard
Chapter 2	About disks and files
Chapter 3	How to install MS-DOS using the Select program How to start the Shell program How to quit MS-DOS
Chapter 4	How to use MS-DOS commands How to print a file
Chapter 5	How to run a program How to create a file
Chapter 6	What the <i>config.sys</i> file does What the <i>autoexec.bat</i> file does How to modify these two files
Terms	About MS-DOS terminology

To learn more about MS-DOS, refer to the *MS-DOS User's Reference*.

1 Learning About MS-DOS

In this chapter you will learn about

- Important MS-DOS terms
- Your personal computer's keyboard

Terms You Should Know

Introducing MS-DOS Terms

When you are introduced to a new or different idea, you must often learn a new set of words to understand the idea. The MS-DOS operating system is no exception. The following pages explain some terms you will need to know so that you can read and use this manual. If you are already familiar with MS-DOS, you may find the *MS-DOS User's Reference* more helpful.

Program

Programs, often called application programs, applications, or software, are series of instructions written in computer languages. These instructions are stored in files and tell your computer to perform a task. For example, a program might tell your computer to alphabetically sort a list of names. Spreadsheets and word processors are other examples of programs.

File

A file is a collection of related information, like the contents of a file folder in a desk drawer. File folders, for instance, might contain business letters, office memos, or monthly sales data. Files on your disks could also contain letters, memos, or data. For example, your MS-DOS master disks contain more than thirty files. Your other disks may contain files that you've created, or that came with the disk.

Filename

Just as each folder in a file cabinet has a label, each file on a disk has a name. This name has two parts: a *filename* and an *extension*. A filename can be from one to eight characters in length, and can be typed in uppercase or lowercase letters. MS-DOS automatically converts filenames to uppercase letters.

Filename extensions consist of a period followed by one, two, or three characters. Extensions are optional, but it's a good idea to use them, since they are useful for describing the contents of a file to you and to MS-DOS. For instance, if you want to be able to quickly identify your report files, you can add the filename extension *.rpt* to each one. Here's an example of a filename with this extension:

```
progress.rpt
```

When you look at the directory on your MS-DOS master disks, you will see many files with the extension *.exe* or *.com*. The extension *.exe* means *executable*, and *.com* means *command*. These extensions tell MS-DOS that the files are programs that can be run. Many files will have other kinds of extensions, such as *.doc* and *.txt*, which might contain text. Another common program file extension is *.bas* for BASIC programs. Some application programs assign filename extensions automatically. For example, Microsoft® Multiplan® assigns the extension *.mp*, and Lotus® 1-2-3 assigns one of three extensions, for instance *.wks* for worksheet files.

Directory

A directory is a table of contents for a disk. It contains the names of your files, their sizes, and the dates they were last modified.

Volume Label

When you use a new disk, you can put a label on the outside of it to help you identify its contents. You can also give each of your disks an internal name, called a *volume label*.

You can look at the volume label on a disk by displaying its directory. Some programs may look at the volume label to see if you are using the correct disk. So make sure that you label your disks. See Chapter 4, "Using Commands", for step-by-step instructions on how to create a volume label for a disk.

Disk Drive

To use the files or programs that are on a floppy disk, you must first insert the disk into a *floppy disk drive*. Floppy disk drives are commonly referred to as the A drive and the B drive. A hard disk drive, normally installed inside your computer, is usually referred to as the C drive. Check your computer manual to see which drive is A and which is B (or C).

Drive Name

A complete *drive name* consists of a *drive letter* and a colon. When using a command, you may need to type a drive name before your filename to tell MS-DOS where to find the disk that contains your file. For example, suppose you have a file named *finances.doc* on the disk in drive B. To tell MS-DOS where to find this file you would type the drive name before the filename:

```
b:finances.doc
```

The Default Drive and the MS-DOS Prompt

If you don't specify a drive name when you type a filename, MS-DOS automatically searches for the file on the disk in the default drive. The *default drive* is where MS-DOS searches first when you type a command. To let you know that it is ready to receive a command, MS-DOS displays a symbol, called a *prompt*, that contains the default drive letter followed by a greater-than sign (>). Following the greater-than sign is the *cursor*, the blinking box or flashing underline that shows where the next character you type will appear. Here's an example of a typical MS-DOS prompt and the cursor:

```
A>_
```

So when your prompt is A>, MS-DOS searches only the disk in drive A (the default drive) for files and programs – unless you tell it to search in another drive.

To change the default drive, you simply type the letter of the desired drive, followed by a colon and press the **ENTER** key. For example, if you will be working primarily with files on drive B, it is easier to change the default drive to B, so that you won't have to type the letter *b*, followed by a colon, with every command and filename.

Changing the default drive

Command

Just as you will run programs to create and update files containing your data, you will also need to run some special programs, called MS-DOS commands, that let you work with entire files.

When you type MS-DOS commands, you are asking the computer to perform tasks. For example, when you use the **diskcopy** command to copy your MS-DOS master disk, you are using a file named *diskcopy.com*, whose task is to copy the files on the MS-DOS disk.

Other MS-DOS commands

- Compare, copy, display, delete, and rename files
- Copy, format, and label disks
- Run your programs, as well as those supplied with MS-DOS
- List directories for disks
- Set the date and time
- Set printer and screen options

You'll learn more about MS-DOS commands in Chapter 4, "Using Commands." But for more detailed descriptions of commands, see the *MS-DOS User's Reference*.

Devices

Whenever you use your computer, you supply the information (input) and expect a result (output). Your computer uses pieces of hardware called *devices* to receive input and send output.

For example, when you type a command, your computer receives input from your keyboard and disk drive, and usually sends output to your screen. It can also receive input from a mouse, or send output to a printer. Some devices, such as disk drives, perform both input and output.

Device Names

Device names are special names given to each device that your computer "knows about". An example of a device name is LPT1, which stands for the first parallel lineprinter connected to your computer.

When you add a new device, such as a mouse, to your computer, you sometimes need to tell MS-DOS about it by setting up (configuring) your computer for that device. Refer to the information that came with your device, or to the *MS-DOS User's Reference*, for more information on configuring your computer for devices.

Error Messages

If you or your computer make a mistake when using a device or MS-DOS command, MS-DOS displays an appropriate *error message*. Error messages apply to general errors (such as misspelling a command) or to device errors (such as trying to use a printer that is out of paper). For a complete list and explanation of each MS-DOS error message (device and general), see the *MS-DOS User's Reference*, Appendix F.

Memory

Memory is the place in your computer where information is actively used. When you run a program, MS-DOS stores that program and the files it uses in the computer's available memory. Some programs and files use more memory than others, depending on how large and complex they are.

Expanded memory is memory beyond 640 kilobytes that uses page-switching technology and that can be used in most personal computers. If you install the expanded memory-board hardware and a special program called the expanded memory manager, then applications that can use expanded memory gain access to this additional memory. When using expanded memory, follow the instructions provided by your memory-board manufacturer for installing the expanded memory manager.

Extended memory is memory beyond one megabyte on a PC-AT or compatible systems. This memory is generally not accessible to DOS-based programs but can be used by memory-disk programs such as RAMDrive. Many popular extended memory boards can be set up to be used as either expanded or extended memory (or both) if the correct software is installed.

The keyboard

Keys You Use with MS-DOS

Now that you've learned about MS-DOS terms, you can learn about the keys you will be using with the MS-DOS operating system.

In addition to the keys you'd find on a typewriter, your computer keyboard has some keys that have special meanings to MS-DOS.

First, note that there are two important differences between a typewriter keyboard and a computer keyboard:



A computer understands the difference between a *one* and a lowercase *L*. Be sure you don't type a lowercase *L* when you mean a *one*.



Capital *O* and *zero* may look alike, but they have different meanings to a computer. Many computers display a zero with a diagonal line (ø) through it. Make sure you type the correct letter or number when you give commands to MS-DOS.

The Enter Key



Press the **ENTER** key after you type commands. When you press the **ENTER** key after typing a command, MS-DOS performs the command.

Moving the cursor

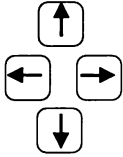
Keys that Move the Cursor



The **SPACEBAR** moves the cursor to the right.



Use the **BACKSPACE** key to correct typing mistakes on the current line. The **BACKSPACE** key deletes characters as it moves the cursor to the left.



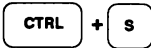
To move the cursor to the left or right *without deleting any characters*, you must use the direction keys. Direction keys move the cursor right, left, up, and down. They do not affect the characters that are displayed. Some programs ignore these keys or do not use them. In these manuals, the direction keys are also referred to as the **RIGHT**, **LEFT**, **UP**, and **DOWN** keys.

Control Key Combinations

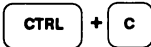
Using the CONTROL key



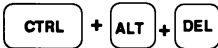
The **CONTROL** key has a special task. It lets you give complex commands to your computer by pressing only two or three keys. You must hold down the **CONTROL** key while you press another key. That is, you use the **CONTROL** key as you would the **SHIFT** key.



When you press the **CONTROL** key and the **s** key at the same time, you can stop the scrolling of the screen display. Then to continue scrolling, press **CONTROL+s** again.



When you press the **CONTROL** key and the **c** key at the same time, you can stop a command.



If you want to restart MS-DOS, press the **CONTROL**, **ALT**, and **DEL** keys at the same time.

In this chapter, you've learned some of the MS-DOS special terms and seen what the keys on your keyboard can do. Now you are ready to go on to Chapter 2, "Learning About Disks, Files, and Directories." There you'll learn about floppy disks and hard disks, about formatting your disks, and about naming your files.

What comes next?

2 Learning About Disks, Files, and Directories

In this chapter you will learn about

- 5.25-inch floppy disks
- 3.5-inch disks
- Hard disks
- Filenames
- Directories

Floppy Disks

About floppy disks

A floppy disk is a flexible, magnetized plastic disk. A double-density floppy disk can store up to 400 single-spaced pages of text. A high-density disk can store about three times that amount. Every floppy disk is enclosed in its own protective cover. The front of this cover is smooth, while the back has visible seams. You should always place labels on the front of the cover, at the top, so that the label doesn't touch the magnetic surface of the disk. It's also a good idea to use a felt-tip pen when writing on labels—a pencil or ballpoint pen can damage the disk if you press too hard.

You should store floppy disks in a safe place, away from dust, moisture, magnetism, and extreme temperatures. Be sure to label each disk you use, since labels help you identify what files are on the disk and remind you that the disk has information stored on it.

Disk Protection

Labels help you keep track of the information on your disks, but you may also need to protect the disks themselves. Some floppy disks are protected, letting you examine information on them without letting you change anything. These are called *write-protected* disks.

Floppy disks can be write-protected in one of two ways. Some have a small piece of tape, called a *tab*, covering a notch on the right side of the disk. You can copy information onto a write-protected disk by first removing the write-protect tab; however, you should consider why the disk was protected *before* you change its contents. After you have copied or changed a write-protected disk it's always a good idea to replace the write-protect tab.

Protecting your disks

If a disk does not have a write-protect notch, it is permanently write-protected. Many application programs come on write-protected disks that protect the files from being destroyed accidentally.

3.5-inch Disks

About 3.5-inch disks

The MS-DOS 4.0 operating system also supports 3.5-inch disks, which, like 5.25-inch floppy disks, are portable magnetic disks. Data on 3.5-inch disks is more densely packed, so depending on the style, a single 3.5-inch disk can store as much (or more) data as a high-density floppy disk.

These smaller disks, sometimes called *microflopies*, have rigid plastic covers with metal shields that guard the disk from dirt and fingerprints. When you place the disk into the disk drive, the computer automatically moves this shield aside to read the disk.

Notice that 3.5-inch disks have a write-protect notch. This notch can be covered with a built-in tab. When the write-protection notch is covered by the tab, no data can be written to the disk.

Be sure to label your 3.5-inch disks and store them in a safe place. As with 5.25-inch floppy disks, extreme temperatures, magnetism, dust, and fingerprints can all harm your data on a disk.

Note

MS-DOS works virtually the same way with both 3.5-inch and 5.25-inch floppy disks. So in this documentation, the term *floppy disk* is used to mean either of these two types of disks.

Hard Disks

In addition to floppy disks, some computers use a hard disk, which can store much more information than a floppy disk. Computers also take less time to find information stored on a hard disk than on a floppy disk. A hard disk is usually built into the computer. Such a system might look like this:

About hard disks

When you store application programs, including MS-DOS, on your hard disk, you should keep a backup copy of the programs on a floppy disk in case the information on the hard disk is accidentally damaged or destroyed. (For more information about making a backup copy of your MS-DOS disks, see Chapter 3, "Getting Started.")

The Format Command

Formatting your disks

Before you can use your new disks for storing information, you must *format* them. You do this with the **format** command, a special program that structures a disk so that MS-DOS can find information on it. The **format** command also checks the disk for defective spots. You can format both floppy and hard disks. But remember that if a disk is not blank, formatting it destroys any data already on the disk.

You will learn more about the **format** command in Chapter 4, "Using Commands."

How to Name Your Files

Naming your files

When naming a file, you may have trouble finding a name that uniquely identifies the file's contents. Dates, for example, are often used in filenames; however, they take up several characters, leaving you with little flexibility. Other common names for files are words like *budget*, *finances*, *analysis*, *report*, etc. These kinds of filenames identify the contents, but leave little room for dates. So the secret is to find a compromise—a point where you can combine a date with a word, creating a unique filename.

The name of a typical MS-DOS file looks like this:

```
customer.lst
```

Notice that the filename was typed in lowercase letters. You can type filenames in uppercase or lowercase letters, even though MS-DOS converts them into uppercase letters. Some more examples of filenames are:

```
budget.86
takeover.bid
june86
finances.doc
schedule.may
```

Many of your filenames will contain only letters and numbers. But you may also use any of the following symbols and letters in your filenames and extensions:

A-Z a-z 0-9 \$ % ' - @ { } ~ ` ! # () &

Warning

Some applications may not let you use all of these symbols. If in doubt, use only letters and numbers.

Invalid Filenames

Although you do have some freedom when naming your files, there are certain names that you may not use, because MS-DOS reserves them for specific devices that your computer uses. These invalid names are *aux*, *clock\$*, *com1*, *com2*, *com3*, *com4*, *con*, *lpt1*, *lpt2*, *lpt3*, *lst*, *nul*, and *prn*. You may use these names as extensions (except for *clock\$*), but remember not to use them to name your files.

Directories

The names of your files are kept in a directory on each disk. The directory also contains information on the sizes of the files, and the dates they were created and updated.

If you want to know what files are on your disk, you can use the **dir** command. This command tells MS-DOS to display all the files in a specific directory on a disk. For example, if your MS-DOS disk is in drive A and you use the **dir** command, the directory display would look similar to this:

```
Volume in drive A is MSDOS400INS
Volume Serial Number is 07E9-3332
Directory of A: \
```

Valid filename characters

Avoiding invalid filenames

The MS-DOS directory

```

COMMAND  COM      37537 06-26-88 12:00a
AUTOEXEC BAT        29 06-26-88 12:00a
CONFIG   SYS        30 16-26-88 12:00a
SELECT   HLP      27994 16-26-88 12:00a
REPLACE  EXE      21431 06-26-88 12:00a
KEYB     COM      14537 06-26-88 12:00a
KEYBOARD SYS     23328 06-26-88 12:00a
FDISK    EXE      70763 06-26-88 12:00a
SYS      COM      11440 06-26-88 12:00a
SELECT   COM       3642 06-26-88 12:00a
FORMAT   COM     30489 06-26-88 12:00a
XCOPY    EXE     17055 06-26-88 12:00a
DISKCOPY COM     10396 06-26-88 12:00a
SELECT   PRT      1330 06-26-88 12:00a
012345   678       109 06-26-88 12:00a
          15 File(s)      12288 bytes free

```

Note

The file sizes and dates you see on your screen may differ from the ones shown here, depending on your version of MS-DOS. Don't worry, though. Such variations do not affect the way you use MS-DOS or the way MS-DOS responds to your commands.

You can also get information about any file on your disk by typing the `dir` command followed by a filename. For example, to display directory information for a file named `schedule`, you could use the following command:

```
dir schedule
```

MS-DOS would respond by displaying the filename `schedule` followed by the file's size in bytes and the date and time it was last changed; for example:

```
SCHEDULE.TXT    3698 8-7-87 4:11p
```

What comes next?

So far you have learned the basic background information that you need in order to use the MS-DOS operating system. In the final three chapters of this guide, you'll learn to make your computer work *for* you, while you build a working knowledge of MS-DOS.

3 Getting Started

In this chapter you will learn

- How to install MS-DOS using the Select program
- How to start MS-DOS Shell
- How to quit MS-DOS
- How to make a backup copies of your MS-DOS disks

How to Install MS-DOS Using the Select Program

The first two chapters in this manual introduced you to the fundamentals of MS-DOS. Now it's time to put your new knowledge to the test. You'll start by installing MS-DOS in your computer using the MS-DOS Select program.

The Select program is a series of screens that prompt you to provide the appropriate information to install MS-DOS in your computer. These screens step you through the choices required to install the operating system into your specific computer.

There are six basic screens that lead you into different screens depending on the information you provide about your system.

The proposed response for each screen of the Select program is highlighted. Use the **UP** or **DOWN** keys to choose an item other than the default selection. In addition, each screen has extensive help information that can be displayed by pressing the **F1** key. Refer to the help information to make a selection for your system.

An overview of the Select program

Preparing to run the Select program

Before you begin the Select program, there are a few things you can do to simplify the installation process.

First, make certain that you know the following information about your system:

- How much memory is available
- What type of printers you have
- Does your system include an expanded memory card

Having these facts ready allows you to respond more quickly to the information the program requires.

Next, prepare the blank disks needed to complete the installation process. The number of disks required for installation depends upon the capacity of the drives you are using. The following table specifies how many blank disks you will need according to your drive type.

Drive Type (Capacity)	Number of Disks (Capacity)
5.25-inch (360 kilobytes)	Four 5.25 (360 kilobytes)
5.25-inch (1.2 megabytes)	Four 5.25 (360 kilobytes)
3.5-inch (720 megabytes)	Two 3.5 (720 kilobytes)
3.5-inch (1.44 megabytes)	One 3.5 (1.44 megabytes)
Hard disk	One 5.25 (360 kilobytes) or one 3.5 (720 kilobytes)

NOTE:

An additional blank disk is required if you are installing MS-DOS Shell.

Last, installing a new system is always a potentially dangerous operation. Always back up all the files on your hard disk before installing MS-DOS 4.0.

Starting the Select program

To start the Select program for either 5.25-inch or 3.5-inch floppy disks, just follow these steps:

1. Make sure your computer is turned off.
2. Take the MS-DOS Install disk out of the protective jacket.

3. Insert this disk into drive A. (Refer to your computer manual for the correct drive.)
4. Close the disk door.
5. Turn on the power for your monitor and your computer.
If you are using 3.5-inch floppy disks, you will skip steps 6 through 9 because the Select program is included on the Install disk. The Welcome screen will simply appear at this point, and you will continue with step 10. If you are using 5.25-inch floppy disks, continue with step 6.
6. Remove the Install disk and insert the Select disk.
7. Press the **ENTER** key.
8. Remove the Select disk and insert the Install disk.
9. Press the **ENTER** key.

The Welcome screen appears. This screen contains information about the number of disks required to install the operating system on either a hard disk or on floppy disks.

10. Press the **ENTER** key.

The Introduction screen appears. This screen contains information about the function keys that you will use to run the Select program. Make a note of the following keys and their functions:

ENTER	Proceeds to next step of program
ESC	Cancels current screen
TAB	Moves cursor to next text entry field
PAGE UP/PAGE DOWN	Scrolls information one page at a time
UP/DOWN	Moves cursor to next item
F1	Displays help information
F3	Exits Select program
F9	Displays key assignments (only when you're using help)
LEFT/RIGHT	Scrolls data horizontally

11. Press the **ENTER** key.

The first screen of the Select program appears. The following section contains more information about the Select program, including some suggestions on how to respond to some of the prompts you will see.

Running the Select Program

The Select program contains six basic screens that prompt you for information about your computer. The selected item on each screen is the default selection. Use the direction keys to select the item that applies to your computer.

The Function and Workspace Screen

Screen 1

This screen lets you allocate memory between MS-DOS and your application programs. Base your allocation on the amount of memory available with your system and the memory requirements of any application programs you may run.

Choose option 1 if your system has 256 kilobytes. This choice lets you allocate a minimum amount of memory to MS-DOS (Function) and a maximum amount to your applications (Workspace).

Choose option 2 if your system has 512 kilobytes. This choice lets you allocate equal memory for MS-DOS and your application programs.

Choose option 3 if your system has more than 512 kilobytes. This choice lets you allocate the maximum amount of memory to MS-DOS, and yet leaves ample memory for your application programs.

The Country and Keyboard Screen

Screen 2

This screen lets you select the appropriate country and keyboard for your computer. You can assign the monetary symbol, decimal separator, date and time format, and keyboard layout applicable to your country and keyboard.

Menus list all available countries and keyboards. If you choose Denmark, Portugal, Norway, or French-speaking Canada, you must set the Code-Page Switching option to Y(es) during the Installation Option Review.

The Select Installation Drive Screen

Screen 3

This screen lets you install MS-DOS onto the hard disk (drive C) or with floppy disks (drive A or B).

If you choose the hard disk, the Specify MS-DOS Location screen appears. Use the options on this screen to specify a new directory or path. If you are installing the Select program to run with a previous version of MS-DOS, you may want to copy the non-system files to a different directory.

The Number of Printers Screen

This screen lets you specify a printer model from a list of printers and choose a printer port. Up to 7 printers may be specified. You may also designate a serial printer port as a parallel port.

If the application programs that you use require a parallel port for printing, you may use this option to name your serial printer with a parallel port name. Do not choose a parallel port name, however, if there is already a parallel printer assigned to that name.

Screen 4

The MS-DOS Shell Option Screen

This screen lets you decide whether or not the Shell should be installed. If it is, then the Shell Start Programs screen appears each time you start your computer.

If you choose not to install the Shell, the MS-DOS prompt (for example, A>) appears when you start your computer. Installing the Shell to run when you start your computer gives you quick and easy access to many basic MS-DOS utilities. You can also “customize” the Shell to install your application programs.

Screen 5

The Installation Options Screen

This screen lets you install MS-DOS using the existing *config.sys* and *autoexec.bat* files, or review and change the selections made for the *config.sys* and *autoexec.bat* files before continuing with installation.

The function and memory values allow MS-DOS and the application programs to run efficiently. Changing any of the set values may result in an increase in the amount of memory used by MS-DOS.

If you choose to continue without reviewing the selections, MS-DOS is copied onto either the hard disk or onto floppy disks. If you choose to review and change the selections, a series of parameter menus appears. The help information for these menus contains examples of valid parameters and memory ranges. In addition, the help information provides guidelines for making the choices that are correct for your system.

Screen 6

Continuing MS-DOS Installation

If you have a hard disk

If you are installing MS-DOS on a hard disk that is not partitioned, you are prompted to provide partition information before installation on the disk can proceed. Since you backed up all existing files before starting the Select program, you may now set up the new partitions, or you may choose to let the Select program partition the disk for you. Refer to the *MS-DOS User's Reference* for additional information on partitioning and formatting your hard disk.

After your hard disk is formatted correctly for version 4.0, the Select program prompts you to complete the installation of MS-DOS by copying the system files onto your hard disk. The choices you made while installing MS-DOS with the Select program were placed into two files: *autoexec.400* and *config.400*. Use these 4.0 files to update your current *autoexec.bat* and *config.sys* files to access the new information. After updating the files, restart your system by pressing the **CONTROL+ALT+DELETE** keys.

If you have floppy disk drives

If you are installing MS-DOS 4.0 on 1.44 megabyte disks, the Select program prompts you through the process of installing MS-DOS on a single disk. When the installation process is complete, you will have a working disk labeled Startup. This is your MS-DOS start-up disk.

However, if you are installing MS-DOS on 720 or 360 kilobyte disks, the Select program prompts you through the process of installing MS-DOS onto three 720 kilobyte disks or five 360 kilobyte disks. For a 720 kilobyte drive, the working disks are labeled

- Startup
- Working
- Shell

You can start MS-DOS from both the Startup and Shell disks. For a 360 kilobyte drive, the working disks are labeled

- Startup
- Working1
- Working2
- Working3
- Shell

Of these five disks, only Startup will start MS-DOS.

An Overview of the MS-DOS Shell

MS-DOS Shell is a graphic extension of MS-DOS that lets you take advantage of the power of MS-DOS with a few keystrokes or a click on a mouse.

When you work with MS-DOS Shell, you are using the graphic extension together with the MS-DOS operating system to perform tasks from menus with a few keystrokes.

MS-DOS Shell lets you perform two kinds of tasks. You can start programs and work with your files. MS-DOS Shell has two main programs: Start Programs and File System. Start Programs lets you select (start) all the other programs available in MS-DOS Shell. That includes File System which lets you perform MS-DOS commands as you work with your files.

When the Shell is active, the Start Programs screen appears. It contains the Main Group of programs you will use. DOS Utilities is one of your choices. It isn't a program, but leads you to another group of programs—utility programs in which you can work with your disks and set the date and time on your computer.

Groups are always readily recognized by the three dots that follow every group name. There are only two groups now, the Main Group (Start Programs) and DOS Utilities, but you can add and change other groups as you want.

For a detailed list of available functions, as well as procedures for using MS-DOS Shell, see the *MS-DOS® Shell User's Guide*.

How to Start the MS-DOS Shell

There are five different ways to start MS-DOS Shell. The method that you used to install the system (based on the capacity of your disk drive) determines how you restart your system and start the Shell. The five methods are described below.

If you installed MS-DOS on your hard disk and chose to install the Shell, remove all disks and press the **CONTROL+ALT+DELETE** keys. This restarts your system and starts MS-DOS Shell.

If you installed MS-DOS on your hard disk and chose not to install the Shell, when you start your computer, type *DOSSHELL* at the MS-DOS prompt.

What Is a Shell?

If you Installed MS-DOS on a hard disk

If you installed MS-DOS on floppy disks

If you installed MS-DOS on a 1.44 megabyte disk, insert the Startup disk in drive A and press the **CONTROL+ALT+DELETE** keys. This restarts your system and starts MS-DOS Shell.

If you installed MS-DOS on 720 kilobyte disks, insert the disk labeled Shell into drive A and press the **CONTROL+ALT+DELETE** keys to start MS-DOS Shell.

If you installed MS-DOS on 360 kilobyte disks, insert the disk labeled Startup into drive A and press the **CONTROL+ALT+DELETE** keys. This restarts the system. Then, remove the Startup disk and insert the disk labeled Shell. Type the word *DOSSHELL*. This starts MS-DOS Shell.

How to Quit MS-DOS

Ending your MS-DOS session

There is no quit command in MS-DOS, but you can end your MS-DOS session easily by following these steps:

1. Make sure that your last command is finished. You should see the MS-DOS prompt (for example, *A>*) on the screen.
2. Remove the floppy disks from the drives, put them back in their protective jackets, and store them in a safe place.
3. Please see the manual that accompanied your hardware for the appropriate procedures for turning off your computer.

How to Make a Backup Copy of Your MS-DOS Disk

In this section, you'll learn how to make a backup copy of your MS-DOS disk (or disks, depending upon how many disks you used during installation). Notice that you are not backing up the original MS-DOS disks, but rather the working disk(s) that you created during the installation process.

MS-DOS comes with a command named **diskcopy** that lets you copy the contents of disks. You need not format your blank disks before you use the **diskcopy** command since **diskcopy** formats the disk for you as it copies.

If you have two floppy disk drives

1. Insert the first disk to be copied into drive A.
2. At the MS-DOS prompt, type the following:

```
diskcopy a: b:
```

3. Press the **ENTER** key.

MS-DOS responds with the following message:

```
Insert SOURCE diskette in drive A:
Insert TARGET diskette in drive B:
Press any key when ready . . .
```

4. Press the **SPACEBAR** to start the **diskcopy** command. The disk copying process takes time, so you'll have to wait awhile.

When the process is complete, MS-DOS asks

```
Copy another? (Y/N)
```

5. Type *N* if you only have one disk to backup, or press *Y* to copy the next disk.

When you complete the disk copying process, label the new disks and write protect them. Then, place the original disks in a safe place, away from dust, moisture, and magnetism. If anything should happen to the copy (or copies) that you have just made, you'll have to use your original disk(s) to make another copy.

If your computer has only one floppy drive, you can still use MS-DOS commands as you would on a system with more than one drive, but you must also specify a drive name when you type a command. By specifying the drive letter, you tell your computer to perform the command on that drive. The drive names A and B then represent the disks that you put into the single drive. In response to your commands, MS-DOS then prompts you to insert the proper disk, as in the following example:

If you have one floppy disk drive

1. Insert the first disk to be copied into drive A.
Use the MS-DOS **diskcopy** command to make your backup copy since **diskcopy** formats the target disk for you as it copies.

2. At the MS-DOS prompt, type the following:

`diskcopy a: b:`

3. Press the **ENTER** key.

MS-DOS responds with the following message

Insert Source diskette in drive A:
Press any key when ready . . .

4. Press the **SPACEBAR**. The copying begins and then the following message is displayed:

Insert TARGET diskette in drive B:
Press any key when ready . . .

5. Remove the source disk, place the blank disk into the drive, and press the **SPACEBAR**. You may need to reinsert the disks for drives A and B several times to complete the copying process.

If you have a hard disk drive

If your computer has a hard disk, you will have used the Select program to copy all the files from the MS-DOS master disks onto the hard disk. Then each time you start MS-DOS, you won't need to use a floppy disk; instead, you'll be able to start MS-DOS directly from the hard disk. When you have copied the MS-DOS files onto your hard disk, the original floppy disks will be your backup copies.

WARNING:

Whenever you format a disk, you destroy its files. It's a good idea to copy any files from your hard disk onto floppy disks before you format the hard disk (to learn how to copy files, see Chapter 4, Using Commands). Once you have formatted your hard disk you *should* never have to format it again.

What comes next?

You now know how to install MS-DOS 4.0 and start the Shell. You've also learned how to start and quit the MS-DOS operating system, and you've used some simple commands to make backup copies of your disks. In the next chapter, you'll learn how to use more MS-DOS commands. As you read about these commands and start to use them, you'll begin to understand how the MS-DOS operating system works. And once you understand the operating system, you'll see what it can do for you.

4 Using Commands

In this chapter you will learn

- How to use file commands
- How to print files
- How to use disk commands

File Commands

You can use several MS-DOS commands to manage your files. Some of the more common commands are **dir**, **copy**, **del**, **rename**, and **print**.

NOTE

The examples in this chapter assume that drive A is the default drive. Also, many of these examples use filenames which are intended for illustrative purposes only – to use these commands, you would substitute the name of a file on the default disk.

The Dir Command

If you want to find out what files are on a disk, you can list its directory by using the MS-DOS **dir** command. For example, to display the directory of the disk in drive B, you would use the following command

```
dir b:
```

You could also display the directory on the hard disk by using the drive letter C instead of B with the **dir** command. If you use the **dir** command without a drive letter, MS-DOS lists the directory of the disk in the default drive.

Using file commands

Listing the MS-DOS directory

Example

Suppose you want to see how many files are in the directory of the MS-DOS disk in drive A. To display this directory you would simply follow these steps:

1. Make sure the MS-DOS disk is in drive A.
2. Make sure the disk drive door (for drive A) is closed.
3. At the MS-DOS prompt, type the following command, then press the **ENTER** key:

```
dir
```

4. If the disk drive door (for drive A) is open when you try to use this command, MS-DOS will display the following error message:

```
Not ready error reading drive A
Abort, Retry, Fail?_
```

To fix this error, you simply close the door for drive A and type **R** (for Retry).

5. MS-DOS then displays the directory. If necessary, you can stop the directory listing from scrolling by pressing **CONTROL+S**. To view the rest of the display, you simply press **CONTROL+S** again.

Your screen should look similar to this:

```
Volume in drive A is MSDOS400INS
Volume Serial Number is 07E9-3332
Directory of A: \

COMMAND  COM      37537  06-26-88  12:00a
AUTOEXEC BAT        29  06-26-88  12:00a
CONFIG   SYS        30  16-26-88  12:00a
SELECT   HLP     27994  16-26-88  12:00a
REPLACE  EXE     21431  06-26-88  12:00a
KEYB     COM     14537  06-26-88  12:00a
KEYBOARD SYS     23328  06-26-88  12:00a
FDISK    EXE     70763  06-26-88  12:00a
SYS      COM     11440  06-26-88  12:00a
SELECT   COM      3642  06-26-88  12:00a
FORMAT   COM     30489  06-26-88  12:00a
XCOPY    EXE     17055  06-26-88  12:00a
DISKCOPY COM     10396  06-26-88  12:00a
SELECT   PRT      1330  06-26-88  12:00a
012345   678        109  06-26-88  12:00a

          15 File(s)      12288 bytes free
```

Note

The file sizes and dates you see on your screen may differ from the ones shown here, depending on your version of MS-DOS. Don't worry, though. Such variations do not affect the way you use MS-DOS or the way MS-DOS responds to your commands.

The Copy Command

If you need to copy files, you can use the copy command to copy one or more files, either on the same disk or from one disk to another. For instance, suppose you need a copy of a file named `sales.doc` that you have on a disk in drive A, and suppose you want to call this new copy *monthly.rpt*.

Example

To copy the `sales.doc` file and call the new copy `monthly.rpt`, you would just follow these steps:

Copying a file

1. Make sure that the disk with the `sales.doc` file is in drive A and that A is the default drive.
2. At the MS-DOS prompt, type the following command:
`copy sales.doc monthly.rpt`
3. Press the ENTER key.

You cannot give the new copy of a file the same name as the original. You can, however, copy a file from one disk to another and keep the same filename. For example, to copy a file from the disk in drive A to the disk in drive B, use the following command:

```
copy a:sales.doc b:sales.doc
```

Note

In the previous example, if A is the default drive (that is, if the prompt is `A>`), you needn't type the letter A, followed by a colon, before the first filename. If you don't specify a new name, the copy will also have the name of the original file. For example, the following commands all produce the same result:

```
copy a:sales.doc b:sales.doc
copy sales.doc b:sales.doc
copy sales.doc b:
```

Again, by substituting the drive letter C for B, you could copy the *sales.doc* file to drive C.

The Delete Command

Just as you may need to make copies of files, you may also need to remove old or unnecessary files to clean up your file system. When you want to erase a file from a disk, you can use the MS-DOS delete (or *del*) command. Remember, though, that the *del* command *permanently* erases the file. To delete an old *sales.doc* file from the disk in drive B, at the MS-DOS prompt you would use the following command:

```
del b:sales.doc
```

You could also delete a file named *sales.doc* from drive C by simply substituting the drive letter C for B.

Example

Deleting a file

Suppose you have an old copy of the *sales.doc* file that you no longer need. To delete this file from the disk in the default drive, you would just follow these steps:

1. Make sure that the disk with the *sales.doc* file is in the default drive.
2. At the MS-DOS prompt, type the following command:

```
del sales.doc
```

3. Press the **ENTER** key. MS-DOS then deletes the *sales.doc* file from the disk.

Note

The *del* command does not work if you type the word *delete*. You can, however, substitute the word *erase* in place of the *del* command.

The Rename Command

Occasionally, you may want to change the name of a file. For example, suppose you have a file named *monthly.rpt* on a disk. When you add other monthly reports to your disk, you may want to change the name of the original file to something more specific. To change the name to *annual.rpt*, for instance, you would use the following command:

```
rename monthly.rpt annual.rpt
```

You can only rename files on the same disk, so you cannot change *a:monthly.rpt* to *b:monthly.rpt* or *c:monthly.rpt*.

Example

Suppose you want to rename a file named *payroll.doc*, on the disk in the default drive, to *salary.doc*. You would simply follow these steps:

1. Make sure that the disk with the *payroll.doc* file is on the disk in the default drive (A).
2. At the MS-DOS prompt, type the following command:

```
rename payroll.doc salary.doc
```

3. Press the ENTER key.

Note

The *rename* command can be abbreviated to *ren*.

The Type Command

If you want MS-DOS to display a file that contains text (often called a text file) on the screen, use the *type* command. For example, say you have created a file named *phone.lst* on the disk in drive A, and you want to check one of the phone numbers. To display the file on the screen, you would use the following command:

```
type a:phone.lst
```

Renaming a file

Example

Displaying a file

Suppose you want to check your employees' salary figures. So you decide to look at a file named *salary.doc* that is on the disk in the default drive. To display the *salary.doc* file, you would just follow these steps:

1. Make sure that the disk with the *salary.doc* file is in the default drive (A).
2. At the MS-DOS prompt, type the following command:

```
type salary.doc/p
```

3. Press the ENTER key.

MS-DOS then displays the *salary.doc* file on the screen.

If the *salary.doc* file is on drive B or C, you could easily type the drive letter, followed by a colon, with the *type* command.

Hints

If the file is too long to fit on the screen, remember that you can use the */p* option to display one screen at a time. When you type *dir/p*, MS-DOS scrolls the text a page at a time. Press any key to see more of the file.

MS-DOS displays only text files on the screen. So if you try to display a program file (one with an extension of *.com* or *.exe*), you will see only strange symbols on the screen.

If you have an application program that creates files, you may need to run the application to view them. For example, if you use Microsoft Multiplan to create a file, Multiplan automatically adds the extension *.mp* to the filename. You would then have to start Multiplan to view the file.

The Print Command

If you have a printer attached to your computer, you can print files with the MS-DOS **print** command. Assume, for example, that you have a file named *invest.mnt* and want to print it on your printer. You could use the following command:

```
print invest.mnt
```

Example

Say you have a file that contains a list of investors and their phone numbers, and suppose you want to print this file and keep it near your phone. The file is named *invest.mnt* and is on the disk in drive B. Drive A is the default drive (A> is the prompt). To print the *invest.mnt* file, you would just follow these steps:

1. Make sure that the MS-DOS disk is in drive A.
2. Make sure that the disk with the *invest.mnt* file is in drive B.
3. Check to see that your printer is on, has paper, and is ready to print.
4. At the MS-DOS prompt, type the following command:

```
print b:invest.mnt
```

5. Press the **ENTER** key.
6. MS-DOS prompts you for the name of the printing device connected to your computer (this name is usually the communications port that the printer cable connects to). Just type the name, or press the **ENTER** key to print to the default printer.

If the master disk is not in drive A, MS-DOS prompts you to insert it in the drive.

Hints

While a file is being printed, you can type other commands to MS-DOS. You can even run other programs or create and modify files. But since printing a file takes a lot of your computer's resources, your tasks may take longer if you try to do them while you are printing a file. So if you have a long file to print, you might schedule the printing for when you plan to be away from your computer.

Printing a file

In addition, if you want to print a file that you've created with an application program, you may also have to use the application program's print command to print the file.

Disk Commands

Using disk commands

This section presents two commands that you use for disks: **format** and **diskcopy**.

The Format Command

When you purchase new disks, they are blank and unformatted. You must format them before MS-DOS can use them. Formatting structures a disk so that MS-DOS can find and store information on it; formatting also checks the disk for defective spots. You can format a disk by using the **format** command.

To format a blank disk in drive B, you would use the following command:

```
format b:
```

Note

If you have only one disk drive, MS-DOS prompts you to insert the disk that you want to format. See Chapter 3, *Getting Started*.

You can also format a blank disk in such a way that some special MS-DOS files are copied onto it during formatting. These files are necessary only if you want to use the disk to start MS-DOS. To format a blank disk in drive B and include these special MS-DOS files, you would use the following command:

```
format b: /s
```

If you don't want to use the disk to start MS-DOS, you don't need to specify the `/s` option when formatting the disk. If you have a disk and don't know whether you can use it to start MS-DOS, put the disk into drive A and press the `CONTROL+ALT+DELETE` key combination. If the disk does not contain the system files, MS-DOS displays an error message.

Example

Suppose you need to create a new data disk to hold some tax records, but you don't want to copy the special MS-DOS files when formatting the disk. To format and label a blank disk (in drive B) *without including the special MS-DOS files*, you simply follow these steps:

Formatting a floppy disk

1. Make sure that the MS-DOS disk is in drive A.
2. At the MS-DOS prompt, type the following command:

```
format b:
```
3. Press the `ENTER` key.
 Your screen should look like this:

```
A>format b:
Insert new diskette for drive B:
and press ENTER when ready . . .
```
4. Insert a blank disk in drive B.
5. Press the `ENTER` key to start the format process.
6. When formatting is complete, MS-DOS displays the following prompt:

```
Volume label (11 characters, ENTER for none) ?
```
7. Type a label that identifies the contents of this disk (for example, **DATA DISK**), and press the `ENTER` key. MS-DOS then asks

```
Format another? (Y/N)
```
7. Type *N* (for No) to exit the **format** program.

In this example, you learned how to format a floppy disk that was in drive B, a floppy disk drive. To format your hard disk, you should follow the instructions in Chapter 3, *Getting Started*.

Now your disk is formatted and ready to use. Be sure to label it on the outside cover, and remember to include the volume label that you used in step 6. The label will remind you that you have formatted the disk, and will help you identify its contents.

Warning

The format program destroys any information already on a disk. It's a good idea to check the directory of a disk before you format it, just to make sure you won't be destroying any important files.

The Diskcopy Command

You may often need to make copies of entire disks instead of individual files. You can do this easily with the MS-DOS **diskcopy** command. To use the **diskcopy** command, you must have

- an MS-DOS disk
- a disk you want to copy
- a blank disk to put the copy on

To copy the contents of a disk in drive A to a disk in drive B, you would use the following command:

```
diskcopy a: b:
```

You can only copy information to disks of comparable media. In other words, you can only copy information from a 5.25-inch floppy disk to a 5.25-inch floppy disk. You cannot copy information from a 5.25-inch floppy disk to a 3.5-inch floppy disk or vice versa.

Note

You cannot use the `diskcopy` command to copy the contents of a floppy disk to or from a hard disk. Instead, you must use the `copy` or `xcopy` commands.

Example

Suppose you want to bring a data disk with you on a business trip, but you don't want to take your original disk because it might get damaged. All you have to do is use the `diskcopy` command to make a copy of the disk. For example, to copy the contents of a disk in drive A to a disk in drive B on a two-drive system, you simply follow these steps:

Copying a floppy disk

1. Put your MS-DOS disk in drive A.
2. At the MS-DOS prompt, type the following command:

```
diskcopy a: b:
```

3. Press the `ENTER` key.
Your screen should look like this:

```
A>diskcopy a: b:
Insert SOURCE diskette in drive A:
Insert TARGET diskette in drive B:
Press any key when ready . . .
```

4. Remove the MS-DOS disk from drive A, replacing it with the disk you want to copy (SOURCE). Then place a blank disk (TARGET) in drive B.
5. Press the `SPACEBAR` to start the `diskcopy` process.
When the disk has been copied, MS-DOS asks

```
Copy another? (Y/N)
```
6. Type `N` (for No) to exit the `diskcopy` program.

What comes next?

If you'd like more detailed information about the commands in this chapter, see your *MS-DOS User's Reference*. But if you are satisfied that you know enough about these commands, you can go on to the next chapter, in which you'll learn how to run some applications with MS-DOS.

5 Using Applications with MS-DOS

In this chapter you will learn some common uses of MS-DOS, such as

- Running application programs
- Creating a file with Edlin

How to Run Application Programs

MS-DOS lets you run many different application programs, including spreadsheets, word processing programs, and graphics packages. These application programs can help you in a number of ways. For instance, they can help you balance a budget, figure income taxes, or manage information, such as stocks, monthly reports, and address lists.

Once you have started MS-DOS, you can run an application program, as follows:

1. If drive A is not the default drive, change the default drive to drive A.
2. Put the application program disk in drive A.
3. Type the name of the application program you want to run.
4. Press the **ENTER** key.

**Starting an application
from a floppy disk**

Example

Suppose you have a word processing application called *Phrase* that you want to use to write a monthly status report.

To start *Phrase* in MS-DOS, you would follow these steps:

1. Make sure that the default drive is drive A by typing the letter A, followed by a colon, and pressing the ENTER key.
2. Put your *Phrase* disk into drive A.
3. Type the name *phrase* (supposing *phrase* is the word used to start the application).
4. Press the ENTER key to start *Phrase*, which you could then use to create, edit, format, or print your status report.

Starting an application from a hard disk

If you want to run an application that is on your hard disk (drive C), follow these steps:

1. Change the default drive to C, the drive that contains the application program.
2. Type the name of the application program you want to run.
3. Press the ENTER key.

Example

Suppose you have a graphics program called *Canvas* stored in drive C, and you want to use it to create a chart showing the current month's sales data.

To start *Canvas* in MS-DOS, you would follow these steps:

1. Change the default drive to drive C by typing the letter C, followed by a colon, and pressing the ENTER key.
2. Type *canvas* (supposing *canvas* is the word used to start the application).
3. Press the ENTER key to start *Canvas*.

You could then use *Canvas* to create your chart.

A Note About Using Application Programs

After quitting some application programs, especially programs that use a lot of memory, you may receive the following error message from MS-DOS:

```
Insert disk with \command.com in drive A  
Press any key to continue . . .
```

This message doesn't mean you have ruined your application program or your computer. It occurs because your application used so much of the computer's memory that it wrote over the MS-DOS *command.com* file. To fix the error, you simply reinsert a disk that contains a copy of *command.com* in the default drive (this *command.com* file must be the same version you used to start MS-DOS). You then press any key when you're ready to continue using your computer.

How to Create a File with Edlin

MS-DOS includes a line-editing program called **Edlin** that lets you create and edit files. **Edlin** is called a line editor because it lets you edit files line by line.

To help you learn how to use **Edlin**, the following section takes you through a sample editing session in which you'll use **Edlin** to create a small file.

Suppose a client asks you to write a catchy advertisement for an electric pencil sharpener, so you decide to create a file named *pencil.ad* on the disk in the default drive. You want the file to contain the following lines:

```
Introducing...  
The X-1000 Automatic Pencil Sharpener  
From Sharpe Office Supplies  
The World Leader in Office Sharpeware
```

Creating a file with Edlin

The following example shows you how to start Edlin, create the *pencil.ad* file, and exit Edlin. All you have to do is follow these steps:

1. Make sure the MS-DOS disk is in drive A.
2. At the MS-DOS prompt, type the following command, then press the **ENTER** key:

```
Edlin pencil.ad
```

Since you are just creating the file, Edlin responds with the following message:

```
New file
*_
```

When you see the asterisk (*), type the letter I (for Insert) and press the **ENTER** key. You will see line number 1 (Edlin uses line numbers to help you edit, but they are not part of your file).

3. Type the following lines. Remember to press the **ENTER** key after each line, including the last line.

```
Introducing...
The X-1000 Automatic Pencil Sharpener
From Sharpe Office Supplies
The World Leader in Office Sharpware
```

Correcting mistakes in Edlin**NOTE:**

If you make a mistake when typing a line, use the **BACKSPACE** key to erase the mistake before you press the **ENTER** key. If you do press the **ENTER** key before correcting the mistake, don't worry about it – you'll learn later how to correct a line.

Your screen should look like this:

```
A>Edlin pencil.ad
New file
*_
1:*Introducing...
2:*The X-1000 Automatic Pencil Sharpener
3:*From Sharpe Office Supplies
4:*The World Leader in Office Sharpware
5:*_
```

4. When you see 5*, press **CONTROL+C** to return to the Edlin prompt (the asterisk).

5. If you made a mistake when typing a line (line 3, for instance), type the number 3, press **ENTER**, and retype the line (remember to press the **ENTER** key at the end of the line).
6. Then at the asterisk (*), type the letter *E* (for End). You will then be returned to the MS-DOS prompt.

You now have a file named *pencil.ad* on the disk in your default drive. If you type the MS-DOS `dir` command, you should see an entry for *pencil.ad*. You can also view this file by using the `type` command as follows:

```
type pencil.ad
```

and then pressing the **ENTER** key.

To learn more about how to use **Edlin**, see the *MS-DOS User's Reference*.

In this chapter, you've learned how to run application programs in MS-DOS, and how to use the line editor, **Edlin**. In the next and final chapter, you'll learn how to set up MS-DOS for *your* needs, for the tasks that you want to do.

What comes next?

6 Setting Up MS-DOS

In this chapter you will learn about

- The `config.sys` file
- The `autoexec.bat` file
- The differences between these two files

This chapter discusses *config.sys* and *autoexec.bat*, two special files that you can use to set up MS-DOS. You'll still be able to use MS-DOS if you don't have these files, but they will help you take greater advantage of the operating system as you run commands and application programs, and as you use devices. In addition to taking greater advantage of the operating system, these special files save you time, by doing tasks for you each time you start MS-DOS.

Special MS-DOS files

The Config.sys File

When you start MS-DOS, it automatically searches for a file named *config.sys* on your system disk. This file contains special commands that let you set up (configure) MS-DOS for use with devices or application programs.

You can use the `dir` command to see whether the *config.sys* file is already on your MS-DOS disk. If the file isn't on the disk, you can use **Edlin** to create it; if it is on the disk, you can use either the **type** command to display it, or **Edlin** to edit it.

Example

A sample *config.sys* file

Although your *config.sys* file should contain the following commands, you shouldn't worry if the file contains more than these two commands:

```
buffers=20
files=20
```

The command **buffers=20** sets the number of buffers, or blocks of memory, that MS-DOS uses to store data. If your directory system is large, you might want to set the **buffers** number higher, for example, to 30.

The second command in the *config.sys* file is **files=20**. This command sets the number of files that MS-DOS can have open at the same time. Programs such as spreadsheets and databases require several files to be open while they are running. If you don't set a value for **files** in your *config.sys* file, MS-DOS assumes a value of 8, which would not be enough open files for a large program like a database.

Note

If you are using MS-DOS with Microsoft Networks, you should set the **files** command to equal 255.

You may also want to add other commands to the file to configure MS-DOS for devices, such as a mouse. You should refer to the manual that came with the device, or to the *MS-DOS User's Reference*, for more information about how to do this.

Creating a config.sys file

If you don't have a *config.sys* file on your MS-DOS disk, simply follow these steps to create one:

1. Type the following command line, then press the **ENTER** key:

```
Edlin config.sys
```
2. At the Edlin asterisk (*) prompt, type the letter **I** (for Insert) and press the **ENTER** key.
3. Now on line 1, type the *config.sys* command **buffers=20**, then press the **ENTER** key.
4. On line 2, type the command **files=20**, then press **ENTER**, followed by **CONTROL+C**.
5. Then at the asterisk (*), type the letter **E** (for End). You will then be returned to the MS-DOS prompt.

MS-DOS performs the commands in the *config.sys* file only when you first start the system; therefore, for your changes to take effect, you must restart MS-DOS after editing this file.

For more information about the *config.sys* file and the *config.sys* commands, see the *MS-DOS User's Reference*, Appendix B, "How to Configure Your System."

The Autoexec.bat File

MS-DOS also searches for a second file when you start your computer. This file is called *autoexec.bat*. It performs any set of commands you would normally give when you start MS-DOS. For example, you might use the file to prepare MS-DOS for running an application program.

If there is an *autoexec.bat* file on the disk when you start MS-DOS, MS-DOS does not automatically prompt you for the time and date at the beginning of your computer session. Therefore, unless you have an installed clock in your computer, it is a good idea to put the **time** and **date** commands in your *autoexec.bat* file. This way, MS-DOS will prompt you for the time and date, and will keep the time and date information current for the directory on your disk.

To see if the *autoexec.bat* file is already on your MS-DOS disk, you simply type the **dir** command. If the file isn't on the disk, you can use **Edlin** to create it, as you did with the *config.sys* file. If the *autoexec.bat* file is on the disk, you can use either the **type** command to display it, or **Edlin** to edit it.

Examples

For a computer with two floppy disk drives, a typical *autoexec.bat* file might contain the following lines:

```
date
time
path=a:
dir
```

In this sample file, the **date** and **time** commands ask you to set the date and time each time you start MS-DOS. The command **path=a:** tells MS-DOS to look for commands or programs on drive A in addition to the default directory. The **dir** command displays the default directory of the disk in the default drive as soon as you start MS-DOS on your computer.

Two sample Autoexec.bat files

The next sample *autoexec.bat* file is for computers with one floppy disk drive and one hard disk drive. It might contain the following lines:

```
date
time
path=c:;a:
prompt=$p$g
dir
```

The commands in this *autoexec.bat* file differ slightly because the file is intended for computers with a hard disk. For instance, the *path* command line now contains *c:*, in addition to *a:*, since when you give a command or start an application, you may want MS-DOS to search two drives, first C, then A.

Another new command in this file is the *prompt=\$p\$g* command, which tells MS-DOS to display the default drive and directory, followed by a greater-than sign (>), as the MS-DOS prompt. This prompt is handy because it reminds you what drive and directory you're in at the moment.

It doesn't matter if your *autoexec.bat* file differs from the ones listed here, but it should at least include the *time* and *date* commands. Also, if you want to start a certain application program every time you start MS-DOS (for example, Microsoft Word), you could include the command to start that application (*word*) at the end of the *autoexec.bat* file.

Once you become more familiar with MS-DOS, you will probably want to vary these commands, or include others. For more information about *autoexec.bat* files or the commands used in these examples, see the *MS-DOS User's Reference*.

How These Special Files Differ

Differences between the special MS-DOS files

MS-DOS uses the *config.sys* and *autoexec.bat* files in different ways because they perform different types of commands. While the *autoexec.bat* file may contain any MS-DOS command or program, the *config.sys* file may contain only a special set of configuration commands.

In addition, you must restart MS-DOS to perform the commands in the *config.sys* file. But to perform the commands in the *autoexec.bat* file, you simply type the word *autoexec*.

Summary

So far you have learned about the basics of the MS-DOS operating system, including some MS-DOS terms, the special keys on the MS-DOS keyboard, and two special MS-DOS files.

You've also learned about:

Using disks, files, and directories

Installing MS-DOS

Starting MS-DOS

Using file and disk commands

Printing files

Running application programs

Using Edlin

Creating *config.sys* and *autoexec.bat* files

If you don't remember everything about these topics, just refer to the appropriate sections to refresh your memory. It's also a good idea to look ahead at the "Terms section." There you can browse through the definitions and familiarize yourself with MS-DOS terminology. Then, once you're confident in your abilities, you can go on to the more advanced *MS-DOS User's Reference*.

In the *MS-DOS User's Reference*, you'll read and learn about multilevel directories, batch files, additional features of Edlin, and much more. Also, if you need a more detailed explanation or example of how to use a command, you can refer to Chapter 3, "MS-DOS Commands." So if this *MS-DOS User's Guide* hasn't answered all your questions, you now know where to look for help.

What comes next?

Terms

The following terms are used in the MS-DOS User's Guide:

****** This abbreviation means *all files in the directory*. The command **copy a:.* b:** means copy all files from the default directory of the disk in drive A to the disk in drive B.

Application software Another name for software, programs, or application programs. Application programs are programs that apply to a specific set of tasks, for example, word processing, spreadsheets, and project management. Software is written in a computer language and consists of a series of instructions that tell the computer to perform tasks.

Autoexec.bat A special MS-DOS file that you use to customize the MS-DOS operating system to your needs. *See also* Config.sys.

Backup disk A copy of any disk you make with the **diskcopy**, **copy**, or **backup** command.

Buffer An area in the computer's memory that MS-DOS uses to store data.

Character A letter, number, or symbol that you type at your keyboard or see on your screen.

Command A short program that tells MS-DOS how to do a specific task.

Config.sys A special MS-DOS file that you use to configure the MS-DOS operating system. The *config.sys* file may contain only a specific set of MS-DOS configuration commands. *See also* Autoexec.bat.

CONTROL key Used in combination with other keys to give MS-DOS special commands such as stop the last command and stop the display from scrolling. Press the CONTROL key at the same time as you press another key.

CONTROL+C A control key combination that stops a command while it is running. *See also* CONTROL key.

CONTROL+S A control key sequence that stops or restarts the scrolling of the screen display. See also **CONTROL** key.

Copy An MS-DOS command that copies one or more files on the same disk, or from one disk to another.

Cursor The lighted shape on the screen that shows where the next character you type will appear. The cursor is usually a blinking line or small box.

Default disk drive The drive where MS-DOS searches for any file-names that you may type. Unless you specify a different drive, MS-DOS looks for files in the default drive. The standard MS-DOS prompt contains the default drive letter. For example, if the prompt is A, then A is the default drive.

Del An MS-DOS command that tells MS-DOS to delete one or more files. A synonym for **del** is **erase**.

Device A piece of hardware that performs a specific function. A printer is an example of a device.

Device errors Errors that occur when one of your computer's devices, usually a disk drive or printer, is not ready or has a problem. When these errors occur, MS-DOS displays a device error message.

Dir An MS-DOS command that means directory. When you type the **dir** command, MS-DOS displays the contents of the disk in the default drive. The command **dir b:** displays the contents of the disk in drive B.

Directory A table of contents for a disk. The directory contains the names of your files, the sizes of the files, and the dates they were created or last modified.

Disk See Floppy disk; Hard disk.

Diskcopy An MS-DOS command that copies disks. **Diskcopy** formats a disk before copying files onto it.

Disk drive A piece of hardware attached to your computer. Typically, a disk drive is either a floppy or hard disk drive. You insert floppy disks in floppy disk drives. Floppy disk drives are commonly referred to as the A drive and the B drive. Hard disks are usually built into the computer and are referred to as the C drive. Your computer manual should tell you how your drives are labeled.

Disk operating system A group of programs that act as a translator between you and your computer. MS-DOS is a disk operating system.

Drive name Consists of a drive letter and a colon. A drive name tells MS-DOS which drive to search for the file. For example, the command type *a:progress.rpt* contains a drive name (*a:*) that tells MS-DOS to look on the disk in drive A for the file called *progress.rpt*.

Editor A program that allows you to manipulate text and data on the computer. Editors allow you to move, add, and delete characters and lines, and to save files. The MS-DOS line editor is called **Edlin**.

Edlin A line-oriented editor that comes with MS-DOS. *See also* Editor.

ENTER key The key you usually press after entering data or text, or after you type an MS-DOS command. On some computers, the **ENTER** key is called the **RETURN** key.

Erase A synonym for the MS-DOS **del** command. *See also* **Del**.

Error messages Messages that appear on the screen after MS-DOS detects a problem while trying to process a command or program. *See the MS-DOS User's Reference* for the appropriate response to each error message.

File A collection of related information. A file on a disk can be compared to a file folder in a desk drawer. For example, a file folder named *friends* might contain the names and addresses of your friends. A file on a disk could contain the same information, and could also be named *friends*. Programs are also stored in files.

Filename A filename can be from one to eight characters in length and can have an extension of up to three characters separated from the filename by a period (.). An example of a complete filename is *progress.rpt*.

Filename extension An addition to a filename. Extensions begin with a period and contain from one to three characters. Most application programs supply their own extensions for files they create. For example, all GW-BASIC files use a filename extension of *.bas*. *See also* **Filename**.

Fixed disk *See* Hard disk.

Floppy disk Used for storing programs and files. In this documentation, the term *floppy disk* includes 3.5-inch as well as 5.25-inch floppy disks.

Format An MS-DOS command that structures blank disks so that MS-DOS can store data on them. You must format every blank disk before it can be used with MS-DOS. **Format** also checks the disk for defective spots.

GW-BASIC A general-purpose computer language. Often, BASIC (or GW-BASIC) is the first computer language that people learn.

Hard disk Sometimes called a fixed disk, one that is built into the computer. A hard disk can store much more information than a floppy disk, and the computer can retrieve information from it faster.

Hardware The equipment that makes up a computer system, not to be confused with the programs or software.

Memory The active part of computer storage used when the computer runs a program or command.

MS-DOS master disk The floppy disk (or disks) on which MS-DOS is distributed. You should always make a backup copy of the master disk (or disks) before you start using MS-DOS on a routine basis.

Operating system A group of programs that translate your commands to the computer, helping you perform such tasks as creating files, running programs, and printing documents. *See also* Disk operating system.

Print An MS-DOS command that prints files on your printer.

Printer A printing device attached to your computer. It lets you print files so that you have a paper copy or printout.

Program A set of instructions, written in computer language, that tells the computer how to perform some task.

Prompt A symbol that usually consists of a default drive letter (usually A, B, or C) and a greater-than sign. An example of the MS-DOS prompt is B>. Other programs will use different prompts. For example, Edlin uses an asterisk (*) as a prompt.

Rename An MS-DOS command that renames files. You can use the abbreviation **ren** in place of the full command name.

RETURN key *See* ENTER key.

Scrolling The movement of text on your screen as it rolls up and off the top of the screen.

Software The programs, routines, or instructions that allow the computer to perform tasks. Some examples of software include: operating systems, word processing programs, and spreadsheet programs.

Type An MS-DOS command that displays files on the monitor.

Volume label An internal name on a disk. You should put a volume label on each of your disks to help you identify them.

Write-protect tab The small tab that covers the write-protect tab on a disk. On 5.25-inch disks, this tab is a removable piece of tape. On 3.5-inch disks, the built-in tab slides up to uncover the write-protect notch. After removing the tab, you can copy information onto the disk. See also Write-protected disk.

Write-protected disk A floppy disk that you can examine information on but cannot change. These disks usually have a write-protect tab. If a disk does not have a write-protect tab, you cannot change any information on the disk. See also Write-protect tab.

Index

- .bas extension 3, 55
- .com extension 3
- .doc extension 3
- .exe extension 3
- .rpt extension 3
- .txt extension 3
- 3.5-inch disk
 - description 14
 - storage capacity 14
 - storing 14
- Abbreviation, *. * 53
- Advanced topics
 - MS-DOS User's Reference 48, 51
- Application program
 - database 48
 - Microsoft Networks 48
 - Microsoft Word 50
 - definition 2, 53
 - running 41-43
 - spreadsheet 2, 48
 - starting from autoexec.bat 50
 - starting from hard disk 42
- Application software 53
- Applications 2
- Arrow keys. See Direction keys; Key
- Asterisk (*)
 - Edlin prompt 45, 48
- Autoexec.bat file
 - creating with Edlin 47, 49-50
 - differences from config.sys file 50
 - hard disk settings 50
 - including time and date commands in 50
 - purpose 47, 49
 - sample file
 - for one floppy disk and one hard disk drive 50
 - for two floppy disk drives 49
 - settings
 - date 49
 - dir 49
 - path 49
- Autoexec.bat file (continued)
 - settings (continued)
 - time 49
- Backing up your MS-DOS disks 26-28
- BACKSPACE key 9, 44
- Backup command 53
- Backup disk, definition 53
- Blank disk 36-39
- Buffer, definition 48, 53
- Buffers command, config.sys
 - description 48
 - setting a value for 48
- Capital letters 3, 8, 17
- Changing a filename 31, 33
- Character 53
- Clock, internal 49
- Colon (:)
 - use of with drive name 5
- Command.com file 43
- Command
 - Backup 53
 - Copy 29, 31-32, 39, 53, 54
 - definition 6, 53
 - Del 29, 32, 54
 - Dir 17-18, 29-30, 49, 54
 - Diskcopy 27, 38-39, 53, 54, 55
 - Erase 55
 - Format 16, 36-38
 - how to stop 9, 53
 - Print 29, 35-36, 56
 - Prompt=\$p\$g 50
 - Rename 29, 33, 56
 - Type 34, 56
- Communications port 35
- Computer language, GW-BASIC 56
- Computer memory 8, 56
- See also Memory

- Config.sys command
 - buffers 48
 - files 48
- Config.sys file
 - commands for 47-48
 - creating with Edlin 47-48
 - differences from autoexec.bat file 50
 - purpose 47
 - restarting after changes 49
 - sample file 48
- Configuration commands, config.sys 48, 50
- Configuring your computer 8, 47
- CONTROL key 9, 53
- CONTROL key sequence
 - CONTROL+ALT+DELETE 9, 37
 - CONTROL+C 9, 45, 48, 54
 - CONTROL+S 9, 30, 34, 54
- Copy command 31-32, 39, 40, 53, 54
- Copying a disk 39
- Copying files 31
- Correcting typing mistakes 44
- Country and Keyboard screen 22
- Country
 - selecting when installing MS-DOS 22
- Creating a config.sys file 48
- Creating a file, Edlin 43-45
- Creating an autoexec.bat file 49-50
- Cursor 5, 8-9, 54
- Database program 48
- Date format 22
- Date prompt 49, 50
- Decimal separator 22
- Default directory 50
- Default disk drive 31, 32, 33, 34, 35, 50
 - changing 5
 - definition 5, 54
- Del command 32, 54
- Deleting a file 32
- Device
 - definition 6, 54
 - device errors 54
 - device name 6-7, 17, 18
 - printing device 35
- Dir command 17-18, 29-30, 49, 54
- Direction keys 9
- Directory
 - contents 18
 - default 50
- Directory (continued)
 - definition 4, 54
 - displaying files 17-18
 - listing 29-30
 - MS-DOS 17-18
- Disk commands 36-39
- Disk drive
 - default 5, 31, 32, 33, 34, 35, 50
 - definition 4, 54
 - floppy 4
 - hard 4
- Disk operating system, MS-DOS 54, 56
- Disk
 - 3.5-inch 14
 - backup 26-28, 53
 - blank 36-39
 - copying 39
 - external label 12, 27, 37, 38
 - floppy 12
 - formatting 16, 36-38
 - hard 15-16
 - labeling 12, 27, 37, 38
 - protecting 13-14, 27, 57
 - volume label 4, 37, 38, 57
 - write-protected 13, 27, 57
 - See also Disk drive; Default disk drive.
- Diskcopy command 27, 38-39, 53, 54, 55
- Displaying a file 33-34
- Double-density floppy disk 12
- DOWN key 21
- DOWN ARROW key 9
- Drive letter 4, 5
- Drive name
 - definition 5, 55
 - floppy disk 22, 54
 - hard disk 22, 54
- Editing a file, Edlin 43-45
- Editor, definition 55
- Edlin
 - asterisk (*) prompt 45, 48
 - creating a config.sys file 47-48
 - creating a file 43-45
 - creating an autoexec.bat file 48-49
 - definition 55
 - description 43
 - editing a file 43-45
 - End command (e) 45, 48

- Edlin (continued)
 - exiting a file 44, 45, 48
 - Insert command (i) 44, 48
 - inserting text 48
 - line editor 43, 55
 - line numbers 44
 - prompt, asterisk (*) 45, 48
- End command (e), Edlin 45, 48
- Ending MS-DOS session 26
- ENTER key 8, 55
- Erase 32, 55
- Erase command. See Del
- Erasing a file 32
- Error message
 - command.com file 43
 - definition 7, 55
 - device error 54
- Errors, device 54
- ESC key 21
- Executable file, .exe 3
- Expanded memory
 - definition 7
 - expanded memory-board hardware 7
- Extended memory
 - definition 7
 - extended memory-board hardware 7
- Extension
 - .bas 3, 55
 - .com 3
 - .doc 3
 - .exe 3
 - .rpt 3
 - .txt 3
 - command 3
 - executable 3
 - filename 3-4, 17, 55
- F1 key 21
- F3 key 21
- F9 key 21
- File commands 29-36
- File
 - autoexec.bat 47, 49-50
 - command 3
 - config.sys 47-48, 49, 50
 - copying 31
 - creating with Edlin 43-45
 - definition 2, 55
 - deleting 32
- File (continued)
 - displaying 33-34
 - executable 3
 - naming 16-17
 - printing 35-36
 - program 34
 - renaming 31, 33
 - text 33-34
- Filename
 - allowed length 3, 55
 - definition 3, 16-17, 55
 - extension 3-4, 17, 55
 - invalid 17
 - renaming 31, 33
 - valid characters in 17
- Files command 48
- Fixed disk. See Hard Disk
- Floppy disk drive
 - definition 4, 50, 54
- Floppy disk
 - backing up, with one disk drive 27-28
 - backing up, with two disk drives 27
 - care of 12-14, 27
 - and MS-DOS installation 19-24
 - description 12-14, 55
 - double-density 12
 - formatting 36-38
 - high-density 12
 - starting applications programs from 41-42
 - starting MS-DOS Shell from 26
 - storage capacity 12
 - write-protected 13-14, 57
- Format command
 - copying special MS-DOS files 37-38
 - description 16, 36-38, 55
 - formatting a floppy disk 36-38
 - options 37
 - volume label 37
 - warning 38
- Formatting a disk 16, 36-38, 55
- Function and Workspace screen 22
- Function keys 8-9, 21
- Further reference
 - MS-DOS User's Reference 48, 51
 - terms, MS-DOS 53-57
- Glossary 53-57
- Greater-than sign (>)
 - used with default drive letter 5, 54, 56

GW-BASIC

definition 56

Hard disk drive

definition 4, 50, 54

Hard disk

installing MS-DOS 20, 22, 28

backing up MS-DOS installation disks 16, 20, 28

backing up programs on 16, 20, 28

description 15-16, 56

drive 4, 54

formatting 22-24

partitioning 24

starting applications programs from 42

starting MS-DOS Shell from 25

High-density floppy disk 12**Input**

definition 6

Insert command (i), Edlin 44, 48**Install disk, MS-DOS 20-21****Installation Options screen 23****Internal name. See Volume label****Invalid filenames 17****Key 8****BACKSPACE 8, 44****CONTROL 9, 53****CONTROL+ALT+DELETE 9, 37****CONTROL+C 9, 45, 48, 54****CONTROL+S 9, 30, 34, 54****direction****DOWN ARROW 9****LEFT ARROW 9****RIGHT ARROW 9****UP ARROW 9****ENTER 8, 21, 55****Function****DOWN 21****ESC 21****F1 21****F3 21****F9 21****LEFT/RIGHT 21****PAGE UP/PAGE DOWN 21****TAB 21****Key (continued)****Function (continued)****UP/DOWN 21****SHIFT 9****SPACEBAR 8****special keys 8****Keyboard****differences between typewriter and computer keyboards 8****keyboard layout 22****keys 8****selecting when installing MS-DOS 22****Label 12, 27, 37, 38****Labeling a disk 27, 37, 38****LEFT ARROW key 9****LEFT/RIGHT key 21****Letters, lowercase 3, 9, 17****Letters, uppercase 3, 9, 17****Line editor, Edlin 43, 55****Line numbers, Edlin 44****Listing a directory 29-30****Lowercase letters 3, 9, 17****Master disk, MS-DOS 28, 56****backing up****floppy disk 27-28****hard disk 16, 20, 28****Memory****allocating memory when installing MS-DOS 22****and application programs 43****buffers 48, 53****definition 7, 56****expanded memory 7****extended memory 7****Message, error 7, 43, 54, 55****Microfloppy disk. See Disk****3.5-inch****Microsoft Networks 48****Microsoft Word 50****Mistakes in typing, correcting 44****Monetary symbol 22****Moving the cursor 5, 8-9, 54****MS-DOS backup disks 26-28****MS-DOS command****Backup 53****Buffers 48****Copy 31-32, 39, 53, 54**

MS-DOS Command (continued)

- definition 7, 53
- Del 32, 54
- Dir 17-18, 29-30, 49, 54
- Diskcopy 27, 38-39, 53, 54, 55
- Erase 32, 55
- Files, config.sys 48
- Format 16, 36-38, 55
- Print 29, 35-36, 56
- Prompt=\$p\$g 50
- Rename 33
- Type 33-34, 56, 58
- MS-DOS directory 17-18
- MS-DOS Install disk 20-21
- MS-DOS Select disk 21
- MS-DOS Shell Option screen 23
- MS-DOS Shell
 - description 25
 - how to start 25-26
- MS-DOS start-up disk 26
- MS-DOS User's Guide
 - summary 51
- MS-DOS User's Reference 48, 51
- MS-DOS Select disk 21
- MS-DOS
 - a quick guide 51
 - advanced topics 51
 - backing up your MS-DOS disks 26-28
 - config.sys commands 47-48
 - default drive letter 5, 54, 56
 - ending session with 26
 - further information
 - MS-DOS User's Reference 48, 51
 - terms 53-57
 - glossary 53-57
 - installing 19-24
 - installing on a hard disk 20, 22, 24, 28
 - keys 8-9
 - line editor, Edlin 43, 55
 - memory, buffer 48, 53
 - open files 48
 - operating system, definition 54, 56
 - prompt 5, 54, 56
 - Select program 19-24
 - Shell 23, 25
 - special files
 - autoexec.bat 24, 47, 49-50
 - config.sys 24, 47-48, 50
 - starting 24-26
 - restarting 9

MS-DOS (continued)

- terms 2-7, 53
- See also MS-DOS command.
- Networks. See Microsoft Networks
- Notational conventions vii
- Notch, write-protect 13-14, 27, 57
- Number of Printers screen 23
- Operating system, MS-DOS
 - configuring 47-50
 - description 54-56
 - setting up 47-50
 - special files
 - sp
 - autoexec.bat 47, 49-50
 - config.sys 24, 47-48, 50
- Output
 - definition 6
- PAGE UP/PAGE key 21
- Path command 49, 50
- Port, communications 35
- Print command 35-36, 56
- Printer port
 - specifying 23
- Printer
 - definition 56
 - specifying 23
- Printing a file 35-36
- Program
 - database 48
 - definition 2, 56
 - Microsoft Networks 48
 - Microsoft Word 50
 - spreadsheet 2, 48
 - See also Application program.
- Programming language, GW-BASIC 56
- Prompt=\$p\$g command 50
- Prompt
 - default drive letter 5, 54, 56
 - definition 56
 - Edlin, asterisk (*) 45, 48
 - MS-DOS 5, 54, 56
- Protecting disks 13, 14, 27, 57

Quitting

Edlin 45, 48

ReferencesMS-DOS User's Reference 48, 51
terms, MS-DOS 51, 53-57**Removing a file 32****Rename command**abbreviation 56
purpose 29, 33, 56**Renaming a file 31, 33**

RESET key See CONTROL+ALT+DELETE

Restarting MS-DOS 9, 49

RETURN key. See ENTER

RIGHT ARROW key 9

Running application programs 41, 43, 49

Sample autoexec.bat file 49-50

Sample config.sys file 48

ScreensCountry and Keyboard 22
Function and Workspace 22
Installation Options 23
MS-DOS Shell Option 23
Number of Printers 23
Select Installation Drive 22
Select program screens 22-23
Shell Start Programs 23
Specify MS-DOS Location 22

Scrolling through text 9

Scrollingstarting 9, 30, 34, 53, 54
stopping 9, 30, 34, 53, 54

Select disk, MS-DOS 21

Select Installation Drive screen 22

Select programand MS-DOS installation 24
Country and Keyboard screen 22
description 19
Function and Workspace screen 22
Installation Options screen 23
installing MS-DOS 19-24
MS-DOS Shell Option screen 23
Number of Printers screen 23
preparing to run 20-21
running 22-24
Select Installation Drive screen 22
starting, with floppy disks 20-21**Select Program (continued)**

using with hard disk 20, 22, 23, 24

Setting up MS-DOS 19-24, 47-48

Shell Start Programs screen 23

Shellinstalling 23
overview 25
starting 25-26

SHIFT key 9

Softwareapplication 53
definition 2, 53, 56
operating system 56
program 2, 56
spreadsheet 56
word processing 56

Source diskette 27, 28, 39

SPACEBAR key 8

Special filesautoexec.bat 24, 47, 49-50
config.sys 47-48
differences between config.sys and autoexec.bat
50

Special keys 8, 9, 21

Specify MS-DOS Location screen 22

Spreadsheet program 48

Start-up disk 24, 26

Starting an application program 41-42

Starting MS-DOS 9, 24-26

Stopping a command 9, 53

Stopping scrolling 9, 30, 34, 53, 54

Storage area

buffer 48

See also Memory.

System prompt, MS-DOS 5, 54, 56

System requirements vi-iii

TAB key 21

Tab, write-protect 13, 57

Target diskette 27, 28, 39

Tasks 6, 7, 25

Terms

MS-DOS 2, 53-57

Text editor, Edlin 43-45

Text file 33-34

Time format 22

Time prompt 49

Type command 33-34, 56

Typing mistakes, correcting 44

UP ARROW key 9
UP/DOWN key 21
Uppercase letters 3, 9, 17

Valid filename characters 17
Volume label 4, 37, 38, 57

Word. *See* Microsoft Word
Write-protect notch 13-14, 27, 57
Write-protect tab 13, 57
Write-protected disk 13, 27, 57
Write-protecting your disks 13-14, 27, 57

Xcopy command 30, 39

Microsoft® MS-DOS®

User's Reference

**Operating System
Version 4.01**

Microsoft Corporation Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software described in this document is furnished under a license agreement or non-disclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or non-disclosure agreement. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose without the express written permission of Microsoft.

© Copyright Microsoft Corporation, 1987, 1988. All rights reserved.

Simultaneously published in the United States and Canada.

Microsoft, the Microsoft logo, MS-DOS, Multiplan, GW-BASIC, and XENIX are registered trademarks of Microsoft Corporation.

IBM is a registered trademark of International Business Machines Corporation.

INTEL is a registered trademark of Intel Corporation.

Lotus is a registered trademark of Lotus Development Corporation.

Contents

Introduction vii

- What is MS-DOS? viii
 - MS-DOS Version 4.0 viii
- Notational Conventions ix
- How to Use This Manual x

1 More About Files and Directories 1

- Before You Start 1
- File Protection 2
- How MS-DOS Keeps Track of Your Files 2
- Multilevel Directories 3
- Paths and Pathnames 7
- Wildcards 9
 - The ? Wildcard 9
 - The * Wildcard 9
- Using Directories 10
 - How to Create a Directory 11
 - How to Change Your Working Directory 11
 - How to Display Your Working Directory 11
 - How to Delete a Directory 12
 - How to Rename a Directory 13

2 About Commands 15

- Types of MS-DOS Commands 15
- Redirecting Command Input and Output 18
 - How to Redirect Your Output 18
 - How to Redirect Input 19
- Filters and Pipes 19
 - Command Pipes 19

3 MS-DOS Commands 21

- Sample Command 23
- Command Options 24
 - More About Options 25
- MS-DOS Commands 26

4 Batch Processing 143

- Why Use Batch Files? 143
- How to Create Batch Files 143
- The Autoexec.bat File 145

	How to Create a Batch File with Replaceable Parameters	149
	How to Use Named Parameters in a Batch File	150
	How to Run a Batch File	151
	How to Use Temporary Files	152
	Batch Processing Commands	153
5	MS-DOS Editing and Function Keys	165
	Special MS-DOS Editing Keys	165
	How MS-DOS Uses the Template	165
	The MS-DOS Editing Keys	166
	How to Use the MS-DOS Template	168
	The MS-DOS Control Characters	170
6	The Line Editor (Edlin)	173
	How Edlin Works	174
	How to Start Edlin	174
	How to End Edlin and Save Your Changes	175
	Special Editing Keys	176
	Using Edlin Commands	182
	Some Tips for Using Edlin Commands	182
	Edlin Command Options	184
	The Line Option	184
	The Question Mark Option	185
	The Text Option	185
	The Edlin Commands	185
7	Link: A Linker	207
	Introduction	207
	Starting and Using Link	208
	Method 1: Using Prompts to Specify Link Files	208
	Method 2: Using a Command Line to Specify Link Files	210
	Method 3: Using a Response File to Specify Link Files	213
	The Lib Files	215
	The Map File	216
	The Temporary Disk File - Vm.tmp	217
	The Link Options	218
	Viewing the Options List	219
	Pausing to Change Disks	219
	Packing Executable Files	220
	Producing a Public-Symbol Map	221
	Copying Line Numbers to the Map File	221
	Preserving Lowercase	222
	Ignoring Default Libraries	223
	Setting the Stack Size	223
	Setting the Maximum Allocation Space	224

Setting a High Start Address	225
Allocating a Data Group	226
Removing Groups from a Program	226
Setting the Overlay Interrupt	227
Setting the Maximum Number of Segments	228
Using DOS Segment Order	228
How Link Works	229
Alignment of Segments	230
Frame Number	230
Order of Segments	230
Combined Segments	231
Groups	232
Fixups	232

8 Debug 235

Introduction	235
How to Start Debug	236
Method 1: Debug	236
Method 2: Command Line	237
Debug Command Information	237
Debug Command Parameters	239
Debug Error Messages	240

Appendix A Instructions for Users with SingleFloppyDiskDriveSystems 277

Appendix B How to Configure Your System 279

What is a Configuration File?	279
Config.sys Commands	280
Sample Config.sys File	280

Appendix C Installable Device Drivers 299

Introduction	299
Ansi.sys	300
ANSI Escape Sequences Used with MS-DOS	300
Display.sys	307
Driver.sys	308
Printer.sys	311
Ramdrive.sys	313
Smartdrv.sys	316

Appendix D Configuring Your Hard Disk (Fdisk) 317

Introduction	317
Checking for a Configured and Formatted Disk	318
How to Start Fdisk	318

iv Contents

How to Create a DOS Partition	319
Selection 1: Create Primary DOS Partition	320
Selection 2: Create Extended DOS Partition	321
Selection 3: Create Logical Drive in the Extended DOS Partition	322
How to Set the Active Partition	323
How to Delete a DOS Partition	324
Selection 1: Delete Primary DOS Partition	325
Selection 2: Delete Extended DOS Partition	325
Selection 3: Delete Logical Drive in the Extended DOS Partition	326
How to Display Partition Data	326
How to Select the Next Fixed Disk Drive	328
Appendix E How to Use Code Pages	329
Introduction	329
National Language Support Codes	330
How to Use Code Pages	331
How to Set Device Code Pages	333
How to Switch Between Code Pages	336
How to List Current Code Pages	336
How to Refresh Lost Code Pages	337
Appendix F MS-DOS Message Directory	337
MS-DOS Messages	337
Index	405

Introduction

What you see before you is a resource manual. It is unlikely that you will ever sit down to read it cover to cover. Rather, you will probably pick it up when you are looking for a specific piece of information. The *MS-DOS® User's Reference* was designed with that in mind.

This manual is written for the person who has some experience with personal computers. To best use this manual, you should be familiar with the introductory information in the *MS-DOS® User's Guide* and know how to start MS-DOS®, how to copy, delete, and rename files, how to make copies of disks, and how to run applications.

Also, you should have already made a backup copy of your MS-DOS master disks and stored the originals in a safe place. If you haven't done so, refer to the *MS-DOS® User's Guide* to learn how to make a backup copy of your disks.

Who should use this manual?

What is MS-DOS?

Introducing MS-DOS

The Microsoft® *MS-DOS operating system* is like a translator between you and your computer. The programs in this operating system allow you to communicate with your computer, your disk drives, and your printer, letting you use these resources to your advantage.

MS-DOS also helps you to manage programs and data. Once you have loaded MS-DOS into your computer's memory, you can compose letters and reports, run programs and languages such as Microsoft GW-BASIC®, and use devices such as printers and disk drives.

MS-DOS Version 4.0

What's new?

This version of MS-DOS has several new features, including:

- Enhancements to the **select** command to facilitate MS-DOS 4.0 installation and configuration. The **select** command is now a full-screen program that takes you through installation and system configuration. **Select** also creates corresponding *config.sys* and *autoexec.bat* files.
- Device driver and file system support of hard disks larger than 32 megabytes.
- Use of expanded memory for MS-DOS buffers and **fastopen** data. Expanded memory support (EMS) requires the use of a Lotus®/Intel®/Microsoft (LIM) 4.0 driver.
- Addition of one new command. The **mem** command displays the amount of used and free memory, a list of allocated and free memory areas, and a list of the programs that are currently loaded.
- Enhanced country support. Four new countries are added to the list of countries supported by MS-DOS.
- **Mode** and **graphics** support for enhanced video adapter modes. MS-DOS 4.0 supports a text mode display interface as well as EGA and VGA display modes. **Mode** and **graphics** also support printers with graphic print modes.
- Improved file system performance for buffer and **fastopen** statements in your *config.sys* file.
- Structured directory tree display via the **tree** command. **Tree** displays the tree structure of a disk, indenting each directory and using block graphics to make the structure more obvious.

- Confirmation on multiple file deletes. A new option for the **delete** command, **/p**, prompts you before a file is actually deleted.
- Various enhancements to existing commands including:

append	fastopen	nlsfunc
attrib	fdisk	replace
backup	graftabl	select
country	graphics	
del	mode	

- Various enhancements to system device drivers including:

<i>ansi.sys</i>	<i>display.sys</i>
<i>driver.sys</i>	<i>printer.sys</i>

Notational Conventions

About this manual

Throughout this manual, the following conventions are used to distinguish elements of text:

bold	Used for commands, options, switches, and literal portions of syntax that must appear exactly as shown
<i>italic</i>	Used for filenames, variables, and placeholders that represent the type of text to be entered by the user
monospace	Used for sample command lines, program code and examples, and sample sessions
SMALL CAPS	Used for keys, key sequences, and certain acronyms

How to Use This Manual

The following table is a quick overview of the topics covered in this manual. For more specific topics, you may have to check the table of contents or the index.

Turn to	If you need to know
Chapter 1	About multilevel directories About paths
Chapter 2	About MS-DOS commands
Chapter 3	How to use MS-DOS commands
Chapter 4	How to make a batch file How to use commands
Chapter 5	About MS-DOS editing keys
Chapter 6	How to use Edlin , the line editor How to use Edlin commands
Chapter 7	How to create executable files with link
Chapter 8	How to debug files with debug
Appendix A	What to do if you have only one disk drive
Appendix B	How to configure your system
Appendix C	About ANSI escape sequences About installing device drivers
Appendix D	How to configure your hard disk
Appendix E	About code pages How to set up your personal computer and peripheral devices with code page support
Appendix F	Error messages and what they mean MS-DOS prompts and what they mean

Other manuals that may help you

These other manuals will also be of interest to you:

<i>MS-DOS® User's Guide</i>	Introduces you to the basics of MS-DOS
<i>MS-DOS® Programmer's Reference</i>	Reference Provides detailed information necessary for programming with MS-DOS

What comes next?

Now that you've seen a brief summary of what topics this manual covers, you're ready to start with Chapter 1, "More About Files and Directories. There you'll learn about some of the more advanced features of MS-DOS."

1 More About Files and Directories

In this chapter you will learn about

- Protecting and keeping track of your files
- Working with multilevel directories
- Using wildcards

Before You Start

Before you read this chapter, you should already know how to start MS-DOS, format and make backup copies of disks, copy and delete files, and run programs. If you are unfamiliar with how to do any of these actions, just refer to the *MS-DOS User's Guide* for more information.

File Protection

Protecting your files

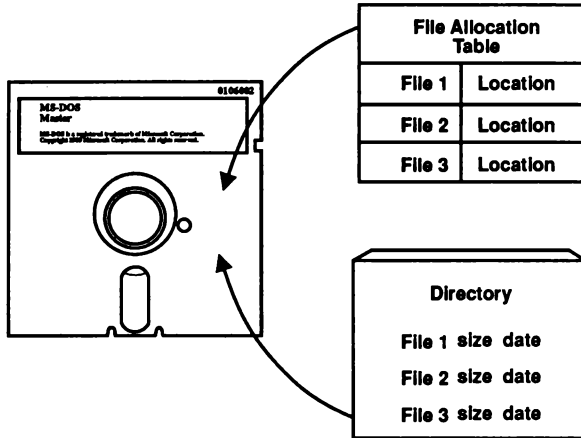
The MS-DOS operating system is a powerful and useful tool for processing personal and business information. As with any computer, this information must be protected, since errors may occur and information may be misused. So if you are doing work that cannot be replaced or that requires a lot of security, you should protect your programs.

You can take simple but effective measures like putting your disks away when you're not using them, or covering the write-protect notch on your program disks. Another way to protect your programs is by installing your equipment in a secure office or work area. Also, if your disks contain valuable information, you should make backup copies of them on a regular basis. For more information on backing up disks, see the **backup** and **restore** commands in Chapter 3, "MS-DOS Commands."

How MS-DOS Keeps Track of Your Files

File Allocation Table

As you learned in the *MS-DOS User's Guide*, MS-DOS stores files in directories. In addition to directories, it uses an area on a disk called the *File Allocation Table*. When you format a disk with the **format** command, MS-DOS copies this table onto the disk and creates an empty directory, called the *root* directory. On each of your disks, the directories store the files, and the File Allocation Table keeps track of their locations. The table also allocates the free space on your disks so that you have enough room to create new files.



These two system areas, the directories and the File Allocation Table, enable MS-DOS to recognize and organize the files on your disks. To check these areas on a disk for consistency and errors, you should use the MS-DOS `chkdsk` command.

For example, to check the disk in drive A, type the `chkdsk` command, followed by `a:`:

In response, MS-DOS displays a status report and any errors it has found, such as files that show a nonzero size in the directory but that really have no data in them.

For an example of such a display and for more information on `chkdsk`, see the description of the `chkdsk` command in Chapter 3, "MS-DOS Commands."

Checking a disk

Multilevel Directories

When there is more than one user on your computer, or when you are working on several different projects, the number of files in the directory can become large and unwieldy. To deal with this large number of files, you may want to keep your files separate from a coworker's, or organize your programs into convenient categories.

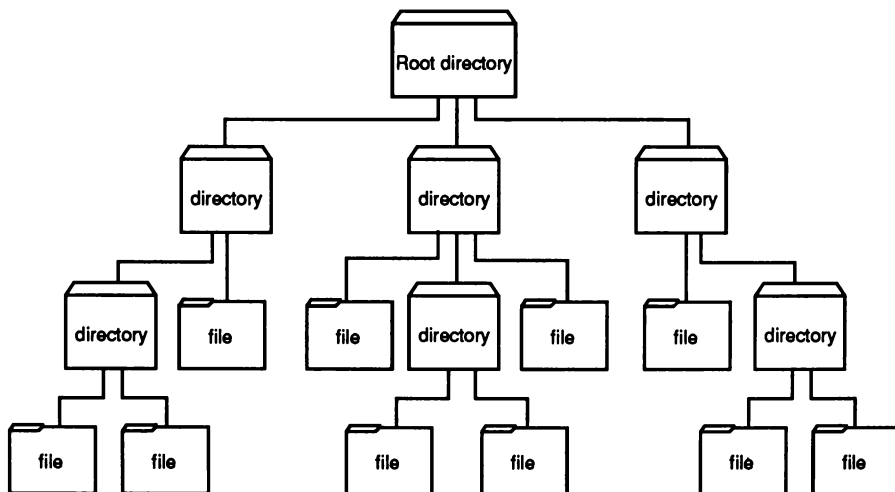
Using multilevel directories

In an office, you can separate and organize files that belong to different people or that relate to specific projects by putting them in different file cabinets. For example, you might put your accounting programs in one file cabinet and your letters in another. You can do the same thing with MS-DOS by putting your files into different directories.

Directories let you group your files in convenient categories. These directories, in turn, may contain other directories (referred to as *subdirectories*). This organized file structure is called a *multilevel* or *hierarchical directory* system.

Note

The maximum number of files or directories that the root directory may contain varies, depending on the type of disk and disk drive you are using. Usually, the maximum number is 112 for a double-sided, double-density, 5.25-inch floppy disk. The maximum number of entries in the root directory of a 1.44 megabyte, 3.5-inch floppy disk is 224. This maximum capacity for a root directory may vary depending upon how the disk is formatted. The number of subdirectories on a disk is not restricted.



The first level in a multilevel directory is the root directory, which is created automatically when you format a disk and start putting files on it. Within the root directory, you can create additional directories and subdirectories.

As you create new directories for groups of files, or for other people using the computer, the directory system grows. And within each new directory, you can add new files or create new subdirectories.

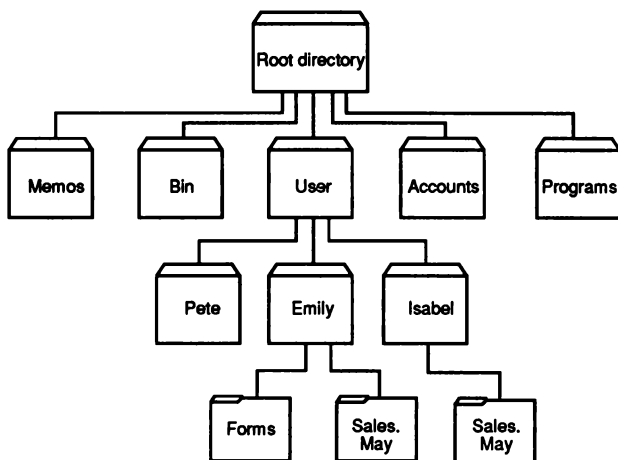
You can move around in the multilevel system by starting at the root and traveling through intermediate subdirectories to find a specific file. Conversely, you can start anywhere within the file system and travel toward the root. Or you can go directly to any directory without traveling through intermediate levels.

The directory that you are in is called the *working directory*. The filenames and commands discussed in this chapter relate to your working directory and do not apply to any other directories in the structure. When you start your computer, you start out in the working directory. Similarly, when you create a file, you create it in the working directory.

Because you can put files in different directories, you and your coworkers can have files with the same names, but with unrelated content. The following figure illustrates a typical multilevel directory structure.

The root directory

Your working directory



In this example, five subdirectories of the root directory have been created. These subdirectories are:

- A directory of external commands, named *bin*.
- A *user* directory containing separate subdirectories for all users of the system.
- A directory containing accounting information, named *accounts*.
- A directory of programs, named *programs*.
- A directory of text files, named *memos*.

As you can see, Pete, Emily, and Isabel each have their own directories, which are subdirectories of the *user* directory. Emily has a subdirectory named *forms*, and both Emily and Isabel have *sales.may* files in their directories, even though Isabel's *sales.may* file is unrelated to Emily's.

This organization of files and directories is not important if you work only with files in your own directory. However, if you work with someone else, or on several projects at once, the multilevel directory system becomes handy. For example, you could get a list of the files in Emily's *forms* directory by typing the following command:

```
dir \user\emily\forms
```

Notice that a backslash (\) separates directories from other directories and files. In the previous example, the first backslash includes the root directory. The use of the backslash alone indicates the root directory. For example, the following command displays a list of the files in the root directory:

```
dir \
```

To find out what files Isabel has in her directory, you would type the following command:

```
dir \user\isabel
```

This command tells MS-DOS to travel from the root directory to the *user* directory to the *isabel* directory, and then to display all filenames in the *isabel* directory.

Paths and Pathnames

When you use multilevel directories, you must tell MS-DOS where the files are located in the directory system. Both Isabel and Emily, for example, have files named *sales.may*, so each would have to tell MS-DOS in which directory her file resides when she wants to use it. This is done by giving MS-DOS a *pathname* to the file.

A *pathname* is a sequence of directory names followed by a filename. Each directory name is separated from the previous one by a backslash (\). A *path* differs from a *pathname* in that it does not include a filename.

The general format of a *pathname* is as follows:

```
[\directoryname] [\directoryname...] \filename
```

What Is a pathname?

A pathname may contain any number of directory names up to a total length of 63 characters. If a pathname begins with a backslash, MS-DOS searches for the file beginning at the root of the directory system. Otherwise, it begins at the working directory and searches along the path from there. Here are two examples:

The pathname of Emily's *sales.may* file is

`\user\emily\sales.may`

The pathname of Isabel's *sales.may* file is

`\user\isabel\sales.may`

When you are in your working directory, you may use a filename and its corresponding pathname interchangeably. Some sample names are:

<code>\</code>	The root directory.
<code>\programs</code>	A directory under the root directory that contains program files.
<code>\user\isabel\forms\1040</code>	A typical full pathname. This one is for a file named <i>1040</i> in the <i>forms</i> directory, which belongs to Isabel
<code>sales.may</code>	A file in the working directory.

Parent directories

A parent *directory* is any directory that contains subdirectories. MS-DOS provides special shorthand notations for the working directory and the parent of the working directory, and automatically creates these two entries whenever you create a directory:

MS-DOS uses the shorthand name “.” to indicate the name of the working directory in all multilevel directory listings.

These two dots are the shorthand name “..” for the working directory's parent directory (one level up). If you type the `dir` command followed by two dots, MS-DOS lists the files in the parent directory of your working directory.

If you type the following command, MS-DOS lists the files in the parent's *parent* directory:

```
dir ..\..
```

Wildcards

If you are using multilevel directories, you will find it easier to search for files on your disks if you use two special characters, called *wildcards*. The wildcard characters are the asterisk (*) and the question mark (?). They are useful in MS-DOS command lines because they give you flexibility when you are specifying paths and files.

The ? Wildcard

A question mark (?) in a filename or filename extension means that any character can occupy that position. The following command, for example, lists all filenames on the default drive that begin with the characters *memo*, that have any character in the next position, that end with the characters *aug*, and that have an extension of *.txt*:

```
dir memo?aug.txt
```

Here are some examples of files that might be listed by the preceding command:

```
MEMO2AUG.TXT
MEMO9AUG.TXT
MEMOBAUG.TXT
```

Using the ? wildcard

The * Wildcard

An asterisk (*) used in a filename or filename extension means that any character can occupy that position *or any of the remaining positions in the filename or extension*. For example, the following command lists all the directory entries on the default drive with filenames that begin with the characters *memo* and that have an extension of *.txt*:

```
dir memo*.txt
```

Here are some examples of files that might be listed by this *dir* command:

```
MEMO2AUG.TXT
MEMO9AUG.TXT
MEMOBAUG.TXT
MEMOJULY.TXT
MEMOJUNE.TXT
```

Using the * wildcard

MS-DOS ignores any filename characters that follow the asterisk wildcard, up to the period that separates the filename from its extension. For example, the command `dir *1.mem` would list *all* the files in the directory with the extension *.mem*, not just those files whose names end with the number *1*.

Important

The wildcard abbreviation `*.*` refers to all files in the directory. This feature can be both powerful and destructive when used with MS-DOS commands. For example, the `del` command followed by the wildcard abbreviation `*.*` deletes all files on the default drive, regardless of filename or extension.

Examples:

Suppose you want to find a certain accounting file but can't remember its exact name. What you can do is list the directory entries for all files named *accounts* in the default directory of drive A (regardless of their filename extensions). To do this quickly, you could just type the following command:

```
dir a:accounts.*
```

Similarly, to list the directory entries for all files with *.txt* extensions or in a directory called *reports* (regardless of their filenames) on the disk in drive B, type the following command:

```
dir b:\reports\*.txt
```

This command is useful if your text files have *.txt* extensions. For example, by using the `dir` command with wildcard characters, you could get a listing of all your text files—even if you don't remember their filenames. For more information on the `dir` command, refer to Chapter 3, "MS-DOS Commands."

Using Directories

The following sections describe how to display, change, and delete any directory. You will also learn how to create directories and subdirectories.

How to Create a Directory

To create a subdirectory in your working directory, use the **mkdir** (make directory) command. For example, to create a new directory named *user* under your working directory, simply type the following command:

```
mkdir user
```

After MS-DOS runs this command, a new directory will exist under your working directory. You can also make directories anywhere in the directory structure by specifying **mkdir** followed by a path. MS-DOS automatically creates the directory.

To create files in the new directory, you can use the MS-DOS line editor, **Edlin**. Chapter 6, “The Line Editor (Edlin),” describes how to use **Edlin** to create and save files. You can also create and save files if you have a word processing program such as Microsoft® Word.

How to Change Your Working Directory

With MS-DOS, it is easy to change from your working directory to a different directory; you simply type the **chdir** (change directory) command followed by a path. For example, to change the working directory to *\user*, you simply type the following:

```
chdir \user
```

You can also specify any path after the command so that you can travel around the directory structure. The following command, for example, puts you in the parent directory of your working directory:

```
chdir ..
```

How to Display Your Working Directory

All commands are executed while you are in your working directory. You can find out the name of the directory you are in by typing the MS-DOS **chdir** command with no path. For example, if your working directory is *\user\pete*, when you type **chdir** and press the **ENTER** key, you will see the following:

```
A:\USER\PETE
```

This message shows your *working drive*, A, plus your working directory, *\user\pete*.

Creating a new directory

Changing directories

Displaying your working directory

A shortcut for the Chdir command

Shortcut

You can also type the letters `cd` for the `chdir` command to save time. For example, the following commands are the same:

```
cd \user\pete
chdir \user\pete
```

If you want to see the contents of the `\user\pete` directory, you can use the MS-DOS `dir` command. The subdirectory might look like this:

```
Volume in drive A has no ID
Directory of A:\USER\PETE

.                <Dir>         08-09-87    10:09a
..               <Dir>         08-09-87    10:09a
TEXT             <Dir>         08-09-87    10:09a
FILE1            TXT          5243 08-04-87    9:30a
4 File(s)                836320 bytes free
```

Notice that MS-DOS lists both files *and* directories in this output. As you can see from the display, Pete has a subdirectory named `text`; the “.” refers to the working directory `\user\pete`; the “..” is short for the parent directory `\user`; and `file1.txt` is a file in the `\user\pete` directory. All these directories and files are on the disk in drive A.

Note

Because files and directories are listed together, you cannot give a subdirectory the same name as a file in that directory. For instance, if you already have a path `\user\pete`, where `pete` is a subdirectory, you cannot create a file named `pete` in the `\user` directory.

How to Delete a Directory

Removing a directory

If you create a directory and decide later that you don't want it any more, you can delete it with the MS-DOS `rmdir` (remove directory) command.

The `rmdir` command lets you delete any directory by specifying its path, but the directory must be *empty* except for the “.” and “..” entries. This prevents you from accidentally deleting files and directories.

To remove all the files in a directory (except for the “.” and “..” entries) type `del` followed by the path of the directory. For example, to delete all files in the `\user\emily` directory, type the following command:

```
del \user\emily
```

MS-DOS prompts you with the following message:

```
Are you sure (Y/N)?
```

If you really want to delete all the files in the directory, type *Y* (for Yes). If not, type *N* (for No) to stop the command.

Now you can use the `rmdir` command to delete the `\user\emily` directory by typing the following command:

```
rmdir \user\emily
```

Shortcut

To save time, you can also use the letters `rd` for the `rmdir` command.

Shortcut: rd

How to Rename a Directory

Renaming a directory

There is no command to rename a directory in MS-DOS. You can, however, rename a directory that has no subdirectories. Suppose, for example, you want to rename the `\user\pete` directory and call it `\user\emily`. To do this you would follow these steps (remember to press the `ENTER` key after each step):

1. To create the new directory, type

```
mkdir \user\emily
```
2. Then to copy the files from the old directory to the new directory, type

```
copy \user\pete\*. * \user\emily
```
3. Now to delete the contents of the old directory, type

```
del \user\pete\*. *
```

(Type *Y* in response to the prompt “Are you sure?”)
4. Finally, to remove the old directory, type

```
rmdir \user\pete
```

What comes next?

In this chapter, you've learned more about files and directories, about wildcards and how to use them, and about some basic MS-DOS commands that help you work with files and directories. In the next chapter, you'll learn about the two kinds of commands, and about using redirection symbols.

2 About Commands

In this chapter, you will learn about

- Internal and external MS-DOS commands
- Redirecting input and output
- Command grouping symbols

Types of MS-DOS Commands

There are two types of MS-DOS commands:

Internal commands

External commands

Internal commands are the simplest, most commonly used commands. When you list the directory on your MS-DOS disk, you cannot see these commands because they are part of a file named *command.com*. When you type internal commands, MS-DOS performs them immediately. This is because they were loaded into your computer's memory when you started MS-DOS. Following is a list of the MS-DOS internal commands:

break	exit	ren (rename)
call	for	rmdir (rd)
chcp	goto	set
chdir (cd)	if	shift
cls	mkdir (md)	time
copy	path	type
ctty	pause	ver
date	prompt	verify
del (erase)	rem	vol
dir		
echo		

What are internal commands?

Using pathnames with internal commands

Some internal commands can use paths and pathnames. Specifically, four commands—**copy**, **dir**, **del**, and **type**—have greater flexibility when you specify a pathname after the command. However, you never specify a path or pathname before any internal command.

The formats of these commands are as follows:

■ **copy** *pathname* *pathname*

If the second *pathname* is a directory (a *path*), MS-DOS copies all the files you specify in the first *pathname* into that directory, as in the following example:

```
copy \user\pete\*.* sales
```

■ **del** *pathname*

If the *pathname* is a directory (a *path*), all the files in that directory are deleted. If you try to delete a path, MS-DOS displays the prompt “Are you sure (Y/N)?.” Type *Y* (for Yes) to complete the command, or *N* (for No) to stop the command. Example:

```
del \user\pete
```

■ **dir** *path*

The following command displays the directory for a specific *path*:

```
dir \user\pet
```

■ **type** *pathname*

You must specify a *pathname* (or *filename*) for this command. MS-DOS then displays this file on your screen in response to the **type** command. Example:

```
type \user\emily\report.nov
```

What are external commands?

Any filename with an extension of *.com*, *.exe*, or *.bat* is considered an *external* command. For example, files such as *format.com* and *diskcopy.com* are external commands. And because all external commands are also files, you can create new commands and add them to MS-DOS. Programs that you create with most languages (including assembly language) will be *.exe* (executable) files. Notice, however, that when you use an external command, you do not need to type its filename extension.

Note

If you have more than one external command with the same name, MS-DOS will run only one of them, according to the following order of precedence: *.com*, *.exe*, *.bat*.

To illustrate this precedence, suppose your disk contains the files *format.com* and *format.bat*. If you were to type the external command *format*, MS-DOS would always run the program *format.com* first, and not run the *format.bat* file at all.

The following external commands are described in Chapter 3, MS-DOS Commands:

append	fdisk	print
assign	find	recover
attrib	format	replace
backup	graftabl	restore
chkdsk	graphics	select
command	join	share
comp	keyb	sort
diskcomp	label	subst
diskcopy	mem	sys
exe2bin	mode	tree
fastopen	more	xcopy
fc	nlsfunc	

Before MS-DOS can run external commands, it must read them into memory from the disk. When you give an external command, MS-DOS immediately checks your working directory to find that command. If it isn't there, you must tell MS-DOS which directory the external command is in. You do this with the **path** command.

When you are working with more than one directory, you may find it more convenient to put all the MS-DOS external commands in one directory. Then, when it needs them, MS-DOS can quickly find the external commands at one location.

Suppose, for example, that you are in a working directory named *\user\prog* and that the MS-DOS external commands are in *\bin*. To find the *format* command, you must tell MS-DOS to choose the *\bin* path, as in the following command, which tells MS-DOS to search in your working directory *and* in the *\bin* directory for all commands:

```
path \bin
```

You need only specify this path once during each computer session. Also, if you want to know what the current path is, you can simply type the **path** command by itself. In response, MS-DOS then displays the working path on the screen.

Using paths with external commands

Using the Path command

You can automatically set your path when you start MS-DOS by including the **path** command in a file called *autoexec.bat*. For more information on the *autoexec.bat* file, refer to Chapter 4, "Batch Processing".

Redirecting Command Input and Output

Usually, MS-DOS receives input from the keyboard and sends its output to the screen. You can, however, redirect this flow of command input and output. For instance, you may want input to come from a file instead of from the keyboard, and you may want output from a command to go to a file or lineprinter instead of to the screen. With redirection symbols, you can also create *pipes* that let the output from one command become the input for another command.

How to Redirect Your Output

Redirecting output

By default, most commands send output to your screen. If you want to change this and send the output to a file, you just use a greater-than sign (>) in your command. For example, the following command displays *on the screen* a directory listing of the disk in the default drive:

```
dir
```

The **dir** command can send this output to a file named *contents* if you type the following:

```
dir > contents
```

If the *contents* file doesn't exist, MS-DOS creates it and stores your directory listing there. If *contents* does exist, MS-DOS replaces what is in the file with the new data.

Appending output

If you want to append your directory or add one file to another (instead of replacing the entire file), you can use two greater-than signs (>>) to tell MS-DOS to append the output of the command (such as a directory listing) to the end of a specified file. For example, the following command appends your directory listing to an existing file named *contents*:

```
dir >> contents
```

If *contents* doesn't exist, MS-DOS creates it.

How to Redirect Input

Often, it's useful to have input for a command come from a file instead of from the keyboard. This is possible in MS-DOS by using a less-than sign (<) in your command. For example, the following command sorts the file *names* and sends the sorted output to a file called *namelist*:

```
sort < names > namelist
```

Redirecting Input

Filters and Pipes

A *filter* is a command that reads your input, transforms it in some way, and then outputs it to your screen. In this manner, the input is “filtered” by the program.

MS-DOS filters include: **find**, **more**, and **sort**. Their functions are as follows:

- find** Searches for text in a file
- more** Displays the contents of a file one screenful at a time
- sort** Alphabetically sorts the contents of a file

You can redirect the output from a filter into a file, or use it as input for another filter by using pipes. The following section explains how filters are piped together.

Command Pipes

If you want to use the output from one command as the input for another, you can *pipe* the commands to MS-DOS. Piping is done by separating commands with the pipe symbol, which is a vertical bar (|). The following command, for example, displays an alphabetically sorted listing of your directory on the screen:

```
dir | sort
```

The pipe sends all output generated by the **dir** command (on the left side of the bar) as input to the **sort** command (on the right side of the bar).

MS-DOS filter commands

Piping commands

Using redirection symbols with pipes

You can also use piping with redirection symbols if you want to send the output to a file. For example, the following command creates a file named *direct.lst* on your default drive:

```
dir | sort > direct.lst
```

The *direct.lst* file now contains a sorted listing of the directory on the default drive.

You can also specify a drive other than the default drive. Suppose, for example, you want to send the sorted data to a file named *direct.lst* on drive B. To do this you could simply type the following:

```
dir | sort > b:direct.lst
```

You can use more than one pipe on a command line. The following command, for instance, sorts your directory, shows it to you one screen at a time, and puts *—More—* at the bottom of your screen when there is more output to be seen:

```
dir | sort | more
```

Since commands and filters can be piped together in many different ways, you will find many uses for them.

What comes next?

In this chapter, you've learned about internal and external MS-DOS commands and how to redirect their input and output. The next chapter lists each of these commands in alphabetical order along with their syntax and comments about their use.

3 MS-DOS Commands

This chapter provides details about all of the MS-DOS commands (the commands are listed in alphabetical order). In this chapter, you will learn

- Which symbols are used to show internal commands, external commands, and commands that won't work over a network
- The structure of the command page
- What command options are
- Which notational conventions are used in this chapter
- How to use each MS-DOS command

Some MS-DOS commands do not work over a computer network. If you try to use these commands over a network, MS-DOS displays the error message "Cannot *command* to a network device", where *command* is the name of the command you typed.

For various reasons, the following commands do not work over a network (see the descriptions of the individual commands for more detailed explanations):

chkdsk	join
diskcomp	label
diskcopy	recover
fastopen	subst
fdisk	sys
format	

The following page shows a sample command page. Each command described in this chapter follows this sample command page format.

Sample command

Sample Command

Purpose:

The "Purpose" section tells you what the command is used for.

Syntax:

commandname [*options*]

where:

commandname is the name of an MS-DOS command. *options* may include *drive.*, *path*, *filename*, *pathname*, *switch*, *argument*, and/or *string*.

Comments:

The "Comments" section describes the command and how to use it. It also explains why the command is useful and explores each of the command's options.

**Sideheads help you find
information fast**

Notes:

The "Notes" section discusses important points related to the command; for instance,

- You can specify a drive and/or path before any command, unless otherwise specified in the "Notes" section.

Examples:

The Examples section gives one or more examples that illustrate how to use the command.

What are command options?

Command Options

Command options give MS-DOS extra information about a command. If you omit options, MS-DOS either prompts you to supply them or it uses a default value. Refer to individual command descriptions in this chapter for default values.

MS-DOS commands use the following syntax:

command [*options*]

Command is an MS-DOS command, and [*options*] is one or more of the following:

Option	Description
<i>drive:</i>	Refers to a disk drive name. You need to specify a drive name only if you are using a file that is <i>not</i> on the default drive. Information transferred between two disks is sent from a <i>source drive</i> to a <i>target drive</i> . You do not need to enter a drive name when you perform an internal command.
<i>path</i>	Refers to a directory name with the following syntax: [\ <i>directory</i>][\ <i>directory</i> ...]\ <i>directory</i>
<i>filename</i>	Refers to a file, and includes any filename extension. The <i>filename</i> option does <i>not</i> refer to a device or drive name.
<i>pathname</i>	A <i>path</i> plus a <i>filename</i> is called a <i>pathname</i> . A <i>pathname</i> uses the following syntax: [\ <i>directory</i>][\ <i>directory</i> ...]\ <i>filename</i>
<i>switch</i>	Controls MS-DOS commands. Switches begin with a slash; for example, /p.

Option	Description
<i>argument</i>	Provides more information to MS-DOS commands. You usually choose between arguments; for example: on or off .
<i>string</i>	Many commands work with <i>strings</i> of text. A string is a group of characters that can include letters, numbers, spaces, and any other characters. Searching for a particular word in a file is a common use of a string.

More About Options

The *MS-DOS User's Reference* uses the following conventions for command options:

Convention	Usage
Notations used for MS-DOS command options	
<i>italics</i>	You must supply the text for any of the variable items shown in italics. For example, when <i>filename</i> appears, you should type the name of your file.
[brackets]	Items in brackets are optional. To include optional information, type only the information within the brackets. Do not type the brackets themselves.
... (ellipsis)	An <i>ellipsis</i> (...) means that you can repeat an item as many times as necessary.
<i>separators</i>	Unless otherwise specified, you must use spaces to separate commands from their options; for example, <pre>rename dull.doc sharpe.doc</pre> With some commands, you may use a semicolon (;), an equal sign (=), or a tab to separate MS-DOS commands from their options. These characters are also known as <i>separators</i>. In this manual, spaces separate commands from their options.

Notations used for MS-DOS command options

The following three pages briefly describe each MS-DOS command.

MS-DOS Commands

The following MS-DOS commands are described in this chapter. Notice that synonyms for commands are in parentheses.

Note

If you have only one floppy disk drive, refer to Appendix A, "Instructions for Users with Single Floppy Disk Drive Systems," before running any of the following commands.

append	Sets a search path for data files.
assign	Assigns a drive letter to a different drive.
attrib	Sets or displays file attributes.
backup	Backs up one or more files from one disk to another.
break	Sets CONTROL+C check.
chcp	Displays or changes the current code page for the command processor <i>command.com</i> .
chdir	Changes directories or displays the working directory (<i>cd</i>).
chkdsk	Scans the disk in the specified drive and checks it for errors.
cls	Clears the screen.
command	Starts the command processor.
comp	Compares the contents of two sets of files.
copy	Copies the specified file(s).
ctty	Lets you change the device from which you issue commands.
date	Displays and sets the date.
del	Deletes the specified file(s) (<i>erase</i>).
dir	Lists the files in a directory.
diskcomp	Compares disks.
diskcopy	Copies disks.
exe2bin	Converts executable (<i>.exe</i>) files to binary format.

exit	Exits the command processor and returns to the previous level.
fastopen	Decreases the amount of time needed to open frequently-used files and directories.
fc	Compares files and displays differences between them.
fdisk	Configures hard disks for MS-DOS.
find	Searches for a specific string of text.
format	Formats a disk to receive MS-DOS files.
graftabl	Loads a table of graphics characters.
graphics	Prepares MS-DOS for printing graphics.
join	Joins a disk drive to a path.
keyb	Loads a keyboard program.
label	Labels disks.
mem	Displays memory information.
mkdir	Makes a directory (md).
mode	Sets operation modes for devices.
more	Displays output one screen at a time.
nlsfunc	Loads country-specific information.
path	Sets a command search path.
print	Prints files.
prompt	Changes the MS-DOS command prompt.
recover	Recovers a bad disk or file.
ren	Renames first file as second file (rename).
replace	Replaces previous versions of files.
restore	Restores backed up files.
rmdir	Removes a directory (rd).
select	Installs MS-DOS on a new floppy disk with desired country-specific information and keyboard layout.
set	Sets one string value to another in the environment, or displays the environment.
share	Installs file sharing and locking.sort

sort	Sorts data forward or backward.
subst	Substitutes a string for a path.
sys	Transfers MS-DOS system files from one drive to the drive specified.
time	Displays and sets the time.
tree	Displays directory and file names.
type	Displays the contents of a file.
ver	Prints the MS-DOS version number.
verify	Verifies all writes to a disk.
vol	Displays the volume label.
xcopy	Copies files and subdirectories.

These commands are described in detail in the following pages.

Append

Append

Purpose:

Sets a search path for data files.

Syntax:

First use only:

append [/x] [/e]

To specify directories to be searched:

append [drive:]path[;[drive:] path]...

To delete appended paths:

append;

To suspend /x processing with the **attrib**, **backup**, **dir**, **replace**, **restore**, and **xcopy** commands:

append [path] [/x[:on]] [/path:on]

or

append [path] [/x:off] [/path:off]

where:

path is the directory that MS-DOS searches for a data file.

Comments:

The **append** command allows you to specify a search path for data files.

The **append** command tells MS-DOS to search the current directory for the data files. If MS-DOS doesn't find the needed data files there, it searches the first directory in the **append** search path. If it still doesn't find them, MS-DOS continues to the second appended directory, and so on. MS-DOS will not search subsequent directories once the data files are located.

The **append** command invoked without the /x switch will cause MS-DOS to search for data files only if open file (INT 21H, Function 0FH), open handle (INT 21H, Function 3DH), or get file size (INT 21H, Function 23H) is used to access the desired files.

Append accepts the following switches:

The Append switches

Switch	Purpose
/x or /x:on	Extends the number of executable files that use the append directories. It allows the executable files that use exec (INT 21H, Function 4BH), search first (INT 21H, Function 11H), and find first (INT 21H, Function 4EH) to use the append directories.
/x:off	Turns off the /x search, find, and execute operations.
/e	Causes appended directories to be stored in the MS-DOS environment. The append command only accepts the /e switch the first time you invoke the command.
/path:on	Indicates that files having drives and/or paths are to be processed.
/path:off	Indicates that files having drives and/or paths are not to be processed.

You can specify more than one *path* to search by separating each with a semicolon (;). If you type the **append** command with the *path* option a second time, MS-DOS discards the old search path and uses the new one.

If you don't use options with the **append** command, MS-DOS displays the current data path.

And if you use the following command, MS-DOS sets the NUL data path:

```
append ;
```

This means that MS-DOS searches only the working directory for data files.

Notes:

- You can use the **append** command across a network to locate remote data files.
- If you are using the MS-DOS **assign** command, you must use the **append** command before **assign**.
- If you want to set a search path for external commands, see the **path** command in this chapter.

- The **append** command tells MS-DOS to search the data path for all files, regardless of their file extensions, only with the following MS-DOS system calls:

Code	Function
0FH	Open File (FCB)
23H	Get (FCB) File Size
3DH	Open Handle
11H	FCB search first (with /x switch only)
4EH	Handle find first (with /x switch only)
4BH	Exec (with /x switch only)

Examples:

Suppose you want to access data files in a directory called *letters* on drive B, and in a directory called *reports* on drive A. To do this, use the following command:

```
append b:\letters;a:\reports
```

Assign

Assign

Purpose:

Assigns a drive letter to a different drive.

Syntax:

assign [x[=]y[...]]

where:

x is the drive that MS-DOS currently reads and writes to.
y is the drive that you want MS-DOS to read and write to.

Comments:

The **assign** command lets you read and write files on drives other than A and B for applications that use only those two drives. You cannot assign a drive being used by another program, and you cannot assign an undefined drive.

Do not type a colon after the drive letters *x* and *y*.

Notes:

- To ensure compatibility with future versions of MS-DOS, you should use the **subst** command instead of **assign**. The following commands, therefore, are equivalent:

```
assign a = c
subst a: c:\
```

- Since the **assign** command disguises the true device type, you should not use **assign**
 - with commands that require drive information (**backup**, **restore**, **label**, **join**, **subst**, **print**)
 - during normal use of MS-DOS, except as required by a program.

Two other commands, **format** and **diskcopy**, ignore drive reassignments.

Examples:

To reset all drives to their original assignments, type the following command and press the **ENTER** key:

```
assign
```

If you want to run an application on a hard disk drive, C, and this application requires you to put your program disk into drive A and your data disk into drive B, you would type this command:

```
assign a=c b=c
```

All references to drives A and B would then go to drive C.

Setting drive assignments

Attrib

Attrib**Purpose:**

Sets or displays file attributes.

Syntax:

attrib [**± r**] [**± a**] [*drive:*]*pathname* [**/s**]

where:

- +r** sets the read-only attribute of a file.
- r** disables read-only mode.
- +a** sets the archive attribute of a file.
- a** clears the archive attribute of a file.

Comments:

The **attrib** command sets read-only and/or archive attributes for files. You may use wildcards to specify a group of files. The attributes of those files matching *filename* are displayed or modified based on the switch selection. **Attrib** doesn't accept a directory name as a valid filename.

The *drive* and *pathname* specify the location of the file or files you want to reference. The **/s** switch processes all subdirectories as well as the path specified.

The **backup**, **restore**, and **xcopy** commands use the archive attribute as a control mechanism. You can use the **+a** and **-a** options to select files that you want to back up with the **backup /m** command, or copy with the **xcopy /m** or **xcopy /a** commands.

Notes:

If an application creates a file that has read *and* write permission, **attrib** forces read-only mode to allow file sharing over a network.

Examples:

To display the attribute of a file called *news86* on the default drive, you would type the following command:

```
attrib news86
```

The following command gives the file *report.txt* read-only permission:

```
attrib +r report.txt
```

Setting a file as read-only prevents you from accidentally deleting or modifying it.

Displaying file attributes

Setting read-only permissions

To remove read-only permission from the files in the `\user\pete` directory on drive B, and the files in any subdirectories, you would type the following command:

```
attrib -r b:\user\pete /s
```

As a final example, suppose you want to give a coworker a disk that contains all files in the default directory of the disk in drive A, except for files with the extension `.bak` that contain old copies of edited files. To copy these files to a disk in drive B, you would type the following:

```
attrib +a a:*.*
```

```
attrib -a a:*.bak
```

and:

```
xcopy a: b: /m
```

or:

```
xcopy a: b: /a
```

If you use the `xcopy /m` switch, `xcopy` automatically turns off the archive bits of the files in drive A as it copies them.

Using the archive switch

Backup

Backup

Purpose:

Backs up one or more files from one disk to another.

Syntax:

backup [*drive1:*][*path*][*filename*] [*drive2:*] [/s]/m[/a] [/f:*size*]
[/d:*date*] [/t:*time*] [/L: [[*drive:*][*path*]*filename*]]

where:

drive1 is the disk drive that you want to back up.

drive2 is the target drive to which the files are backed up.

Comments:

The **backup** command can back up files on disks of different media (hard disks and floppy disks). **Backup** also backs up files from one floppy disk to another, even if the disks have a different number of sides or sectors.

The **backup** command accepts the following switches:

The Backup switches

Switch	Purpose
/s	Backs up subdirectories.
/m	Backs up only those files that have changed since the last backup.
/a	Adds the files to be backed up to those already on the backup disk. It does not erase old files on the backup disk. This switch will not be accepted if files exist that were backed up using backup from MS-DOS version 3.2 or earlier.
/f: <i>size</i>	Causes the target disk to be formatted if it is not already. For this switch to function, the MS-DOS format command must be accessible by the current path. The values that <i>size</i> can take are described in the next section.
/d: <i>date</i>	Backs up only those files that you last modified on or after <i>date</i> .
/t: <i>time</i>	Backs up only those files that you last modified at or after <i>time</i> .
/L: <i>filename</i>	Makes a backup log entry in the specified file. If you do not specify <i>filename</i> , backup places a file called <i>backup.log</i> in the root directory of the disk that contains the files being backed up.

Size	Possible Values	
160K single-sided 5.25-inch disk	160, 160K, 160KB	Disk sizes
180K single-sided 5.25-inch disk	180, 180K, 180KB	
320K double-sided 5.25-inch disk	320, 320K, 320KB	
360K double-sided 5.25-inch disk	360, 360K, 360K	
720K double-sided 3.5-inch disk	720, 720K, 720KB	
1.2M double-sided 5.25-inch disk	1200, 1200K, 1200KB, 1.2, 1.2M, 1.2MB	
1.44M double-sided 3.5-inch disk	1440, 1440K, 1440KB, 1.44, 1.44M, 1.44MB	

A backup log file uses the following format:

Backup log files

- The first line lists the date and time of the backup.
- A line for each backed-up file lists the filename and number of the backup disk on which the file resides.

If the backup log file already exists, **backup** appends the current entry to the file.

You can also use the backup log file when you need to restore a particular file from a floppy disk, but you must specify which disk to restore so that the **restore** command does not have to search for files. The **restore** command always puts a file back in the same place from which it was backed up. **Backup** displays the name of each file as it is backed up.

You should label and number each backup disk consecutively to help you restore the files properly with the **restore** command. If you are sharing files, MS-DOS lets you back up only those files to which you have access.

Notes:

- The **backup** utility will not back up the system files *io.sys*, *msdos.sys*, or *command.com*.
- You cannot use an old version of the **restore** command (MS-DOS 3.2 or earlier) for files backed up with the MS-DOS 4.0 **backup** command.
- Unless you use the **/a** switch, **backup** erases the old files on a backup disk before adding new files to it.

- You should not use the **backup** command if the drive you are backing up has been assigned, joined, or substituted with the **assign**, **join**, or **subst** commands. If you do, you may not be able to restore the files with the **restore** command.
- The **backup** program returns the following exit codes:

Exit codes

Code	Function
------	----------

0	Normal completion
1	No files were found to back up
2	Some files not backed up due to sharing conflicts
3	Terminated by user
4	Terminated due to error

You can use the batch processing **if** command for error processing that is based on the *errorlevel* returned by **backup**.

Examples:

Backing up files

Suppose Emily wants to back up all the files in the `\user\emily` directory on drive C to a blank, formatted disk in drive A. To do this, she would type

```
backup c:\user\emily a:
```

Break

Break

Purpose:

Sets CONTROL+C check.

Syntax:

break [on]

or

break [off]

Comments:

Depending on the program you are running, you may use CONTROL+C to stop an activity (for example, to stop sorting a file). Normally, MS-DOS checks to see whether you press CONTROL+C while it is reading from the keyboard or writing to the screen or printer. If you set **break** to **on**, you extend CONTROL+C checking to other functions, such as disk reads and writes.

Notes:

Some programs may set themselves to respond to CONTROL+C at any time. Setting **break** does not affect these programs.

Examples:

To check for CONTROL+C only during screen, keyboard, and printer reads and writes, type the following:

```
break off
```

To find out how **break** is currently set, type the **break** command and press the ENTER key.

CONTROL + C checking

Chcp

Chcp**Purpose:**

Displays or changes the current code page for the command processor *command.com*.

Syntax:

chcp [*nnn*]

where:

nnn is the code page to start.

Comments:

The **chcp** command accepts one of the two prepared system code pages as a valid code page. An error message is displayed if a code page is selected that has not been prepared for the system.

If you type the **chcp** command without a code page, **chcp** displays the active code page and the prepared code pages for the system.

You may select any one of the prepared system code pages defined by the **country** command in *config.sys*.

The following are valid code pages:

Value	Code Page
437	United States
850	Multilingual
860	Portuguese
863	Canadian-French
865	Nordic

Any program that you run after starting a new code page will use the new code page. Programs that started before the new code page will still use the original code page.

To see what the current code page setting is, you simply type

chcp

Chcp messages

MS-DOS will respond with a message similar to the following:

Active code page: 850

Prepared system code pages: 850 437

If you select a code page that is not prepared for the system, MS-DOS displays a message like the following:

```
Code page 850 not prepared for system
Active code page: 437
Prepared system code pages: 437 865
```

If a device (screen, keyboard, printer) is not prepared for a code page, MS-DOS displays the following error message:

```
Code page 850 not prepared for device xxx
```

Notes:

- In the preceding example, the `chcp` command will still change the active code page even if the selected code page is not prepared for a device. If you want to change the active code page to the original code page, you must reissue the `chcp` command with the original code page selected.

Example:

To set the code page for the current screen group to 863 (Canadian-French), you would type the following command:

```
chcp 863
```

Chdir (change directory)**Chdir****Synonym:**

cd

Purpose:

Changes a directory to a different path; displays the working directory.

Syntax:

chdir[*path*]

Comments:

The **chdir** command changes your working directory to the directory you specify. A shorthand notation for the **chdir** command is **cd**. Thus, either of the following commands will change your current directory to the directory called *primetim*:

```
chdir \primetim
```

```
cd \primetim
```

To display the name of your working directory, simply type:

```
cd
```

Shortcuts

There are two shortcuts you can use when you want to change your directory to a parent directory or subdirectory of your working directory. To illustrate, suppose you have a directory called *specials* that has a subdirectory called *sponsors*. So to change your working directory to *\specials\sponsors*, you would type

```
cd \specials\sponsors
```

If your working directory is *specials*, you can type the following command to change to the *\specials\sponsors* directory:

```
cd sponsors
```

Then, if you wanted to change your working directory back to the parent directory, *\specials*, you could type this command:

```
cd ..
```

Type `cd \` to return to the “root” directory. The root directory is the highest-level directory on your computer and is usually the directory that you see when you start MS-DOS.

Examples:

If you use `chdir` without a path, you can display the name of your working directory. For example, if your working directory is `\user\pete` on drive B, and you type the command, `chdir b:`, then press the `ENTER` key, MS-DOS displays the following:

```
b:\user\pete
```

**Displaying the name of
your working directory**

Chkdsk (check disk)

Chkdsk**Purpose:**

Scans the disk in the specified drive and checks it for errors.

Syntax:

chkdsk [*drive:*][*pathname*] [*/f*] [*/v*]

Comments:

The **chkdsk** command shows the status of your disk. You should run **chkdsk** occasionally on each disk to check for errors. If you do run **chkdsk** on a disk and any errors are found, **chkdsk** displays the error messages, followed by a status report.

A typical status report might look like this:

```
Volume MSDOS created 10-19-1988 9:54 a
Volume Serial Number is 0D41-10FA

362496 bytes total disk space
347136 bytes in 38 user files
15360 bytes available on disk

1024 bytes in each allocation unit
354 total allocation units on disk
15 available allocation units on disk

655360 total bytes memory
538688 bytes free
```

If you type a filename after **chkdsk**, MS-DOS displays a status report for the disk and for the individual file.

The **chkdsk** command accepts the following switches:

Switch	Purpose
/f	Fixes errors on the disk. Because repairs usually involve changes to the disk's File Allocation Table that may cause a loss of information, you are prompted with a confirmation message similar to the following: <pre>Ten lost clusters found in 3 chains. Convert lost chains to files (Y/N)?</pre> If you type <i>Y</i> (for Yes), each lost chain becomes a file with the name <i>filennnn.chk</i>. When chkdsk is complete, you can check the <i>.chk</i> files to see if they

The Chkdsk switches

Switch	Purpose
	contain any data you need. If you type <i>N</i> (for No) the contents of the lost clusters are not saved to files.
	If you do not specify the <i>/f</i> switch, chkdsk does not correct errors that it finds in your directory. However, it does display messages about files that need to be fixed.
<i>/v</i>	Displays the name of each file in each directory as it checks the disk.

If you specify the */f* switch, **chkdsk** will show an error if it finds any open files on the disk. If you do not specify the */f* switch and there are open files, **chkdsk** may give results that make it seem like there are lost clusters on the disk. This happens when the File Allocation Table has yet to be updated regarding open files. If **chkdsk** reports a large number of clusters as lost, you should consider repairing the disk.

Notes:

- **Chkdsk** does not correct errors on a disk unless you specify the */f* switch. For more information on **chkdsk** errors, refer to the specific error message in Appendix F, "MS-DOS Message Directory."
- **Chkdsk** doesn't work on drives used in the **subst** or **join** commands.

Examples:

If you want to save a **chkdsk** status report for future use, you can redirect the output from **chkdsk** to a file, *status*, by typing the following:

```
chkdsk a:>status
```

The errors are then sent to the specified file. Remember, though, not to use the */f* switch when you redirect **chkdsk** output.

If **chkdsk** finds errors on the disk in drive A and you want to try to correct them, type the following command:

```
chkdsk a:/f
```

Chkdsk now tries to correct any errors it finds on the disk in drive A, prompting you for further information when necessary.

Saving a Chkdsk status report

Cls

Cls**Purpose:**

Clears the screen.

Syntax:

`cls`

Comments:

The `cls` command clears your screen, leaving only the MS-DOS prompt and a cursor.

Examples:

Clearing your screen

Clearing your screen

You may find it more comfortable to work with a "clean slate." If you want to start a new process with a clear screen, type

`cls`

Command

Purpose:

Starts the command processor.

Syntax:

command [*drive:*][*path*][*ctty-dev*] [*/e:nnnnn*][*/p*][*/c string*]

where:

ctty-dev allows you to specify a different device (such as AUX) for input and output.

Comments:

This command starts a new command processor (the MS-DOS program that contains all internal commands).

When you start a new command processor, you also create a new command environment. This new environment is a copy of the old, parent environment. However, you can change the new environment without affecting the old one.

The command processor is loaded into memory in two parts: *transient* and *resident*. Some applications write over the transient memory part of *command.com* when they run. When this happens, the resident part of the command processor looks for the *command.com* file on disk so that it can reload the transient part.

Drive: and *path* tell the command processor where to look for the *command.com* file if it needs to reload the transient part into memory.

The following switches are accepted by **command**:

Switch	Purpose
<i>/e:nnnnn</i>	Specifies the environment size, where <i>nnnnn</i> is the size in bytes, ranging from 160 to 32,768. MS-DOS rounds this number up to the next logical paragraph boundary. The default value is 160 bytes.
<i>/p</i>	Keeps the secondary command processor in memory and does not automatically return to the primary command processor.
<i>/c string</i>	Tells the command processor to perform the command or commands specified by <i>string</i> and then to return automatically to the primary command processor.

Command

The Command switches

If *nnnn* is less than 160 bytes, MS-DOS defaults to 160 bytes and displays the following message:

```
Invalid environment size specified
```

If *nnnn* is greater than 32,768 bytes, MS-DOS displays the same message, but defaults to 32,768 bytes.

Notes:

For more information about *ctty-dev*, see the *ctty* command in this chapter.

Examples:

Starting a new command processor

The following command tells the MS-DOS command processor to do three things:

- Start a new command processor under the current program
- Run the *chkdsk* command to check drive B.
- Return to the first command processor

```
command /c chkdsk b:
```

The *shell* command can also be used with *command* to increase space. If you want to increase to 256 kilobytes, enter the following:

```
shell=command.com /p /e:256
```

Comp

Purpose:

Compares the contents of two sets of files.

Syntax:

comp [drive:][*pathname1*] [drive:][*pathname2*]

Comments:

The **comp** command compares one file or set of files (*pathname1*) with a second file or set of files (*pathname2*). These files can be on the same drive or on different drives. They can also be in the same directory or different directories.

The two sets of files you want to compare can have the same path and filenames-provided they are on different drives. If you type only a drive for the second option, **comp** assumes that *pathname2* is the same as *pathname1*. You can use wildcards (* and ?) to specify the pathnames.

If you don't type the pathname options or if you omit the *pathname2* option, **comp** prompts you for them. If either option contains only a drive or a path with no filename, **comp** assumes the filename is *.*.

If the files you want to compare are on a different disk than **comp**, type the **comp** command with no options. When **comp** prompts you for the pathname options, you can insert the correct disk and type the filenames to be compared.

As **comp** proceeds, it displays the paths and names of the compared files. A message appears if **comp** cannot find a file matching the *pathname2* option, or if a directory path is invalid. If no file matches the *pathname1* option, **comp** prompts you for both the pathname options again.

During the comparison, a message appears for any location in the two files that contains mismatching information. The message indicates the offset into the files of the mismatching bytes and the contents of the bytes themselves (all in hexadecimal notation). The message has the following format:

```
Compare error at OFFSET XXXXXXXX
file1 = XX
file2 = XX
```

Comp

Comparing files on different disks

In this format, *file1* is the first filename typed; *file2* is the second filename typed. After ten unequal comparisons, **comp** stops comparing and displays the following message:

```
10 Mismatches - ending compare
```

If the file sizes are different, **comp** displays the following message:

```
Files are different sizes, do you wish to  
continue (Y/N)?
```

You can either continue the comparison or end it. If you choose to continue, **comp** compares the files until it reaches the end of the shorter file.

After a successful comparison, **comp** displays the following message:

```
Files compare OK
```

After the comparison of the two files ends, **comp** proceeds with the next pair of files that match the two pathname options, until no more files can be found that match the *pathname1* option. Then **comp** displays the following message:

```
Compare more files (Y/N)?
```

You can now compare two more files, or end the comparison. If you want to compare two more files, type *Y* (for Yes). **Comp** prompts you for two new path options.

For all file comparisons, **comp** first ensures that both files include end-of-file (CONTROL+Z) marks. If they do not, **comp** displays this message:

```
EOF mark not found
```

Examples:

In the following example, **comp** compares each file with the extension *.asm* in the current directory on drive C with each file of the same name (but with an extension of *.bak*) in the current directory on drive B:

```
comp c:*.asm b:*.bak
```

Copy

Purpose:

Copies one or more files to another location. This command also appends files.

Syntax:

To copy files:

```
copy [drive:][pathname1] [drive:][pathname2][[/v]][/a][/b]
```

or

```
copy [drive:][pathname1] [/v] [/a][/b][drive:][pathname2]
```

To append files:

```
copy pathname1 + pathname2 [...] pathnameN
```

Comments:

If you do not specify *pathname2*, the copy is created in the working directory on the disk in the default drive. This copy has the same name, creation date, and creation time as the original file (*pathname1*). If the original file is on the default drive and you do not specify *pathname2*, the **copy** command quits (you are not allowed to copy a file to itself), and MS-DOS displays the following error message:

```
File cannot be copied onto itself
0 File(s) copied
```

The **copy** command accepts the following switches:

Switch	Purpose
/v	Causes MS-DOS to verify that the sectors written on the target disk are recorded properly.
/a	Lets you copy ASCII files. This switch applies to the filename preceding it, and to all remaining filenames in the command, until copy encounters another /a or /b switch. This switch tells the command processor to read until the end-of-file mark.
/b	Lets you copy binary files. This switch applies to the filename preceding it, and to all remaining filenames in the command, until copy encounters another /a or /b switch. This switch tells the command processor to read the number of bytes specified by the file size in the directory.

Copy

The Copy switches

If MS-DOS cannot verify a write, it displays an error message. Although recording errors rarely occur with the **copy** command, the **/v** switch lets you verify that critical data has been correctly recorded; it also causes the **copy** command to run more slowly, because MS-DOS must check each entry recorded on the disk.

Notes:

- The **copy** command switches **/a** and **/b** perform differently depending on whether they are placed following the source filename or the target filename.

Using the **/a** and **/b** switches

When used with a source filename,

- /a** Causes the file to be treated as an ASCII (text) file. Data in the file is copied up to but not including the first end-of-file mark (in **Edlin**, this is **CONTROL+Z**). The remainder of the file is not copied.
- /b** Causes the entire file to be copied, including any end-of-file marks.

When used with a target filename,

- /a** Causes a end-of-file character to be added as the last character of the file; for example:

```
copy memo.doc /a letter.doc
```
- /b** Does not add an end-of-file character; for example:

```
copy billing.asm /b billing2.asm
```

When you are combining files, the default switch is always **/a**.

- Do not try to append files if one of the source filenames has the same name or extension as the target.
- If you want to copy all of a directory's files and subdirectories, you should use the **xcopy** command. Refer to the **xcopy** command in this chapter for more information.

Copying files and subdirectories

Examples:

To copy a file called *animal.typ* from your working drive and directory to a directory on drive C called *bigcats*, type

```
copy animal.typ c:bigcats
```

The **copy** command also lets you append files. To do this, simply list any number of files as options to **copy**, each separated by a plus sign (+), and then specify a target file to send the combined files to; for example:

```
copy intro.rpt + body.rpt + b:sum.rpt report
```

This command combines files named *intro.rpt*, *body.rpt*, and *sum.rpt* (from drive B), and places them in a file called *report* on the default drive. When you are appending files, the target file is created with the current date and time. If you omit the target file, MS-DOS combines the files, and stores them under the name of the first specified file.

You can also combine several files into one by using wildcards; for example:

```
copy *.txt combin.doc
```

This command takes all files with an extension of *.txt* and combines them into one file named *combin.doc*.

In the following example, each file that matches **.txt* is combined with its corresponding *.ref* file. The result is a file with the same filename but with the extension *.doc*. Thus, *file1.txt* is combined with *file1.ref* to form *file1.doc*, *xyz.txt* with *xyz.ref* to form *xyz.doc*, and so on:

```
copy *.txt + *.ref *.doc
```

The following **copy** command combines all files matching **.txt* and all files matching **.ref*, into one file named *combin.doc*:

```
copy *.txt + *.ref combin.doc
```

Copy compares the filename of the source file with the filename of the target. If they are the same, that one input file is skipped, and MS-DOS prints the error message, "Content of destination lost before copy." Further joining proceeds normally. For example, the following command appends all **.txt* files (except *all.txt*) to *all.txt*:

```
copy all.txt + *.txt
```

This command will not produce an error message.

Appending files

Combining files

Ctty

Ctty**Purpose:**

Lets you change the device from which you issue commands.

Syntax:

ctty device

where:

device specifies the device from which you are giving commands to MS-DOS.

Comments:

Ctty is useful if you want to change the device on which you are working. In this command, the letters **tty** represent your terminal; that is, your computer's screen and keyboard.

Notes:

There are many programs that do not use MS-DOS for input, output, or either. These programs send input directly to the hardware on your computer. The **ctty** command has no effect on these programs; it affects only programs that use MS-DOS.

Examples:

Changing the command input and output device

The following command moves all command I/O (input/output) from the current device (the console) to an AUX port, such as another terminal:

```
ctty aux
```

The next command moves I/O back to the console screen and keyboard:

```
ctty con
```

**Changing the command
input and output device**

Date

Date

Purpose:

Displays or sets the date.

Syntax:

date [*mm-dd-yy*]

Comments:

You can change the date from your terminal or from a batch file. (MS-DOS does not automatically display a prompt for the date if you use an *autoexec.bat* file, so you may want to include a **date** command in that file.) MS-DOS records this date in the directory when you create or change a file.

Remember to use only numbers when you type the date; allowed numbers are

mm = 1-12

dd = 1-31

yy = 80-79 or 1980-2079

The date, month, and year entries may be separated by hyphens (-) or slashes (/). MS-DOS is programmed to change months and years correctly, whether the month has 28, 29, 30, or 31 days.

It is possible for you to change the *mm-dd-yy* format in which the date is displayed and entered. The **country** command in the *config.sys* file allows you to change the date format to the European standard *dd-mm-yy*. For more information on the *config.sys* file, see Appendix B, "How to Configure Your System."

Notes:

- This command sets the internal clock in your computer, if one exists.
- The format *mm-dd-yy* may vary if you are using a code page other than the one for the United States. For more information about international date formats, see Appendix E, "How to Use Code Pages."

**Displaying the
current date****Examples:**

If you simply type the **date** command, MS-DOS displays the following message:

```
Current date is weekday mm-dd-yy  
Enter new date (mm-dd-yy):_
```

where *weekday* is the day of the week (for example, Tuesday).

If you do not want to change the date shown, press the **ENTER** key. Or you can type a particular date after the **date** command, as in the following example:

```
date 3-9-88
```

In this case, the "Enter new date:" prompt does not appear after you have pressed the **ENTER** key.

Del (Erase)

Del (Erase)

Purpose:

Deletes the specified files.

Syntax:

```
del [drive:]pathname/p  
or  
erase [drive:]pathname /p
```

Comments:

The `/p` switch causes **del** or **erase** to prompt you before a deletion actually occurs. If this switch is specified, each file is displayed and you are prompted with *filename*, Delete (Y/N)? You can either confirm the delete with a Y or cancel the delete with an N.

The **del** command lets you use the `*` and `?` wildcards to delete more than one file at a time. While convenient, this method of deleting files can be dangerous, so use wildcards cautiously.

If you type `del *.*`, this tells MS-DOS that you want to delete all the files in the working directory. MS-DOS displays the prompt "Are you sure?". If you type *Y* (for Yes) in response, MS-DOS deletes all files in the working directory.

To delete all the files in another directory, type the **del** command followed by the directory name.

Warning

Once you have deleted a file from your disk, you cannot recover it.

Examples:

The following command deletes a file named *vacation*:

```
del vacation
```

If you have two files named *vacation.feb* and *vacation.apr*, you can delete them both with the following command:

```
del vacation.*
```

Using wildcards with Del

Deleting a file

Dir (Directory)

Dir**Purpose:**

Lists the files in a directory.

Syntax:

dir [*drive:*][*pathname*][*/p*][*/w*]

Comments:

The **dir** command, typed by itself, lists all directory entries in the working directory on the default drive. **Dir** also lists the drive's volume label and volume serial number.

If you include a drive name (such as *b:*) with the **dir** command, all entries in the default directory of the disk in the specified drive are listed.

The **dir** command accepts the following switches:

Switch Purpose

/p	Selects page mode, causing the directory display to pause once the screen is filled. To resume scrolling the display, press any key.
/w	Selects wide display and causes MS-DOS to display only file-names and not other file information. The wide display lists up to five files per line.

Dir lists all files with their size in bytes and the time and date of last modification.

Notice that the following **dir** commands are equivalent, since you can use the wildcards **?** and ***** in the *pathname* option:

This command	Is equivalent to
dir	dir *.*
dir filename	dir filename.*
dir .ext	dir *.ext

Notes:

If the **country** command in the *config.sys* file is set to a country other than the United States, the directory date and time formats may differ. For more information on the *config.sys* file, see Appendix B, "How to Configure Your System."

The Dir switches

Examples:

If your directory contains more files than you can see on the screen at one time, type the following:

```
dir /p
```

This command displays the directory one screen at a time. As one screen fills up, you can press any key to see the next screen of the directory listing.

Diskcomp

Diskcomp**Purpose:**

Compares the contents of the disk in the source drive to the disk in the target drive.

Syntax:

diskcomp [*drive1*:] [*drive2*:] [/1] [/8]

where:

drive1 is the source drive.

drive2 is the target drive.

Comments:

Diskcomp performs a track-by-track comparison of the disks. It automatically determines the number of sides and sectors per track based on the format of the source disk.

The **diskcomp** command accepts the following switches:

Switch	Purpose
--------	---------

- | | |
|----|--|
| /1 | Causes diskcomp to compare just the first side of the disk, even if the disks and drives that you are using are double-sided. |
| /8 | Causes diskcomp to compare just the first 8 sectors per track, even if the disks contain 9 or 15 sectors per track. |

If you specify only one drive, **diskcomp** uses the default drive as the target drive. If you specify the same drive as the source and target, **diskcomp** does a comparison using one drive, and prompts you to insert the disks as appropriate.

If all the tracks are the same, **diskcomp** displays the message:

Compare OK

If the tracks are not the same, **diskcomp** displays a "Compare error " message that includes the track number and side number (0 or 1) where it found the mismatch.

If the target disk is not the same type as the disk in the source drive, **diskcomp** displays the following message:

Drive types or diskette types not compatible

The Diskcomp switches

When **diskcomp** completes the comparison, it prompts you with the following message:

```
Compare another diskette (Y/N)?_
```

If you type *Y* (for Yes), **diskcomp** prompts you to insert the proper disks and does the next comparison. If you type *N* (for No), **diskcomp** ends. If the disk in the default drive does not contain MS-DOS and you end **diskcomp**, you'll receive the following message:

```
Insert disk with COMMAND.COM in drive A
and strike any key when ready
```

Diskcomp does not work on network drives, and you cannot use it with assigned, joined, or substituted drives. If you attempt to use the **diskcomp** command with these types of drives, it displays an error message.

Notes:

- When comparing a disk with a backup disk that you made with the **copy** command, you may receive the "Compare error" message, even if the files on the disks are identical. This is because the **copy** command duplicates the information, but doesn't necessarily place it in the same location on the target disk. In this case, you should use the **fc** command to compare individual files on the disk. For more information, see the **fc** command later in this chapter.
- **Diskcomp** ignores the volume serial numbers in its comparison.
- The **diskcomp** command returns the following exit codes:

Code	Function
0	Compared OK The disks compared exactly.
1	Did not compare The disks were not the same.

Diskcomp exit codes

Code	Function
2	CONTROL+C error The user terminated with CONTROL+C
3	Hard error An unrecoverable read or write error occurred-did not compare.
4	Initialization error There is not enough memory-invalid drives or command line syntax.

You can use the batch processing **if** command to perform error processing based on the *errorlevel* returned by **diskcomp**.

Examples:

Comparing two disks

If your computer has only one floppy disk drive, drive A, and you want to compare two disks, you can simply type the following command:

```
diskcomp a:
```

MS-DOS prompts you to insert each disk, as required.

Diskcopy

Diskcopy

Purpose:

Copies the contents of the floppy disk in the source drive to a formatted or unformatted floppy disk in the target drive.

Syntax:

diskcopy [*drive1:*] [*drive2:*] [/1]

where:

drive1 is the source drive.

drive2 is the target drive.

Comments:

Drive1 and *drive2* may be the same. If you omit the drive options, MS-DOS prompts you for the drives. If the target disk is not formatted, **diskcopy** formats it with the same number of sides and sectors per track as the source disk.

The /1 switch allows you to copy only one side of a disk.

The Diskcopy switch

Warning

The **diskcopy** command works only with floppy disks. You cannot use **diskcopy** with a hard disk.

If you omit both options, MS-DOS performs a single-drive copy operation on the default drive. If you omit just the second option, MS-DOS uses the default drive as the target drive. In either case, though, **diskcopy** destroys the contents of the target disk.

Diskcopy prompts you to insert the source and target disks at appropriate times and waits for you to press any key before continuing.

After copying, **diskcopy** then prompts you with the following message:

Copy another diskette (Y/N)?_

If you type *Y* (for Yes), MS-DOS prompts you to insert source and target disks, and performs the next copy on the drives that you originally specified.

To end the **diskcopy** process, type *N* (for No).

Because disk space is not allocated sequentially, disks that have had a lot of files created and deleted on them become fragmented. As a result, the first free sector found by **diskcopy** becomes the next sector allocated, regardless of its location on the disk.

A fragmented disk can slow down finding, reading, or writing a file. To prevent further fragmentation, you should use either the **copy** command or the **xcopy** command to copy your disk, instead of using the **diskcopy** command. Because the **copy** and **xcopy** commands copy files sequentially to a disk, the new disk will not be fragmented.

The following command, for example, copies all files from the disk in drive A to the disk in drive B:

```
xcopy a:.* b:
```

Diskcopy figures out the number of sides to copy, based on the source drive and disk.

Notes:

The **diskcopy** command returns the following exit codes:

Diskcopy exit codes

Code	Function
0	Copied successfully
1	Non-fatal read/write error An unrecoverable but non-fatal read or write error occurred.
2	CONTROL+C error The user entered CONTROL+C to terminate diskcopy .
3	Fatal hard error Diskcopy was unable to read the source disk or format the target disk.
4	Initialization error There is not enough memory-invalid drives or command line syntax.

You can use the batch processing **if** command to perform error processing based on the *errorlevel* returned by **diskcopy**.

If the source disk has a volume serial number, **diskcopy** creates a new volume serial number for the target disk. The target disk's unique serial number is displayed when the **diskcopy** operation is complete.

Examples:

To copy the disk in drive A to the disk in drive B, use the following command:

```
diskcopy a: b:
```

Diskcopy prompts you to insert both disks and press any key to begin copying.

Copying a disk

Exe2bin

Exe2bin**Purpose:**

Converts *.exe* (executable) files to binary format.

Syntax:

exe2bin [*drive:*]*pathname1* [*drive:*]*pathname2*

where:

pathname1 is the input file.

pathname2 is the output file.

Comments:

This command converts *.exe* (executable) files to binary format. If you do not specify an extension for *pathname1*, it defaults to *.exe*. The input file is converted to a *.bin* file format (a memory image of the program) and placed in the output file (*pathname2*).

If you do not specify a drive name, **exe2bin** uses the drive of the input file. Similarly, if you do not specify an output filename, **exe2bin** uses the input filename. And finally, if you do not specify a filename extension in the output filename, **exe2bin** gives the new file the extension *.bin*.

Some restrictions do apply when you use the **exe2bin** command: the input file must be in valid *.exe* format produced by the linker; the resident, or actual, code and data part of the file must be less than 64K bytes; and there must be no STACK segment.

With **exe2bin**, two kinds of conversions are possible, depending on whether or not the initial CS:IP (Code Segment:Instruction Pointer) is specified in the *.exe* file:

- If the CS:IP is not specified in the *.exe* file, **exe2bin** assumes you want a pure binary conversion. If segment fixups are necessary (that is, if the program contains instructions requiring segment relocation), the **exe2bin** command prompts you for the fixup value. This value is the absolute segment at which the program is to be loaded. The resulting program will be usable only when loaded at the absolute memory address specified by your application. The command processor will not be able to load program.

- If the CS:IP is 0000:100H, `exe2bin` assumes that the file will run as a `.com` file with the instruction pointer set at 100H by the assembler statement `ORG` (the first 100H bytes of the file are deleted). No segment fixups are allowed, since `.com` files must be segment relocatable; that is, they must assume the entry conditions explained in the Microsoft Macro Assembler manuals. Once the conversion is complete, you may rename the output file with a `.com` extension. The command processor will then be able to load and execute the program in the same way as the `.com` programs supplied on your MS-DOS disk.

Exit

Exit**Purpose:**

Exits the *command.com* program (the command processor) and returns to a previous level, if one exists.

Syntax:

exit

Comments:

If you use the MS-DOS **command** program to start a new command processor, you can use the **exit** command to return to the old command processor. Also, while running an application program, you can exit to the MS-DOS command processor, and then return to your program.

For more information about command processors, see the **command** earlier in this chapter.

Examples:

If you start a new command processor by typing the following command:

```
command c:\
```

you can then return to the previous command processor by typing the **exit** command:

```
exit
```

Fastopen

Fastopen

Purpose:

Decreases the amount of time needed to open frequently-used files and directories.

Syntax:

fastopen [*drive*:[*n*][...]] /x
 or
fastopen [*drive*:[*n,m*]][...] /x or
fastopen [*drive*:[*n*][*m*]][...] /x

where:

n is the number of files that **fastopen** will work with. The range is 10-999.

m is the number of file extent entries for files on the drive specified. If omitted, this feature is not provided. The range is 1-999.

Comments:

Fastopen tracks the location of files and directories on a disk for fast access. Access to files in a complex directory structure can be time consuming. If you run applications that use several files (such as a database application), the time to open and close files noticeably degrades your computer's performance.

Every time a file or directory is opened, **fastopen** records its name and location. **Fastopen** also keeps a map of the file's physical location on the drive. Then, if a file or directory recorded by **fastopen** is reopened, the access time is greatly reduced.

Fastopen works only on hard disks, and will not work over a network. You can use **fastopen** with up to four hard disks at one time. For each hard disk, **fastopen** will track *n* files or directories, where *n* ranges from 10 to 999. The default is 10.

The /x switch indicates that the cache will be in (LIM 4.0) expanded memory. For best results with this switch, use the **select** command's default values.

Notes:

- You can invoke the **fastopen** command only once. If you want to change the **fastopen** settings, restart MS-DOS.
- **Fastopen** needs approximately 48 bytes of memory for each file (*n*) that it tracks and 16 bytes of memory for each file extent entry (*m*).

How Fastopen works

The Fastopen switch

Examples:

If you want MS-DOS to track the location of up to 100 files on drive C, you could type the following:

```
fastopen c:=100
```

To disable the name cache for drive C, and instead set up an extent cache having 10 entries, you would type the following:

```
fastopen c:=(,10)
```

Fc**Purpose:**

Compares two files, or two sets of files, and displays the differences between them.

Syntax:

For ASCII comparisons:

```
fc [/a] [/c] [/L] [/Lb n] [/n] [/t] [/w] [/nnnn][drive:]pathname1
[drive:]pathname2
```

For binary comparisons:

```
fc [/b] [/nnnn][drive:]pathname1 [drive:]pathname2
```

where:

pathname1 is the first file that you want to compare.

pathname2 is the second file that you want to compare.

Comments:

Fc matches the first file against the second and reports any differences between them.

The nine switches that you can use with the fc command are described as follows:

Switch	Purpose
/a	Abbreviates the output of an ASCII comparison. Instead of displaying all the lines that are different, fc displays only the lines that begin and end each set of differences.
/b	Forces a binary comparison of both files. Fc compares the two files byte-by-byte, with no attempt to resynchronize after a mismatch. The mismatches are printed as follows: xxxxxxx: yy zz (where xxxxxx is the relative address from the beginning of the file of the pair of bytes). Addresses start at 00000000; yy and zz are the mismatched bytes from <i>pathname1</i> and <i>pathname2</i> , respectively. The /b switch is the default when you compare .exe, .com, .sys, .obj, .lib, or .bin files.

Fc

The Fc switches

Switch	Purpose
<code>/c</code>	Causes the matching process to ignore the case of letters. Fc then considers all letters in the files as uppercase letters.
<code>/L</code>	Compares the files in <code>ASCII</code> mode. This switch is the default when you compare files that do not have extensions of <code>.exe</code> , <code>.com</code> , <code>.sys</code> , <code>.obj</code> , <code>.lib</code> , or <code>.bin</code> .
<code>/Lb</code>	Sets the internal line buffer to <i>n</i> lines. The default length of the internal buffer is 100 lines. Files that have more than this number of consecutive, differing lines will abort the comparison.
<code>/n</code>	Displays the line numbers on an <code>ASCII</code> comparison.
<code>/t</code>	Does not expand tabs to spaces. The default is to treat tabs as spaces to 8- column positions.
<code>/w</code>	Causes fc to compress white space (tabs and spaces) during the compression. If a line contains many spaces or tabs in a row, these characters are considered a single white space. Notice that although fc <i>compresses</i> white space, it does not ignore it. The two exceptions are beginning and ending white spaces in a line, which are ignored.
<code>/nnnn</code>	Specifies the number of lines that must match after fc finds a difference between files. If the number of matching lines in the files is less than this number, fc displays the matching lines as differences.

How Fc reports differences

Fc reports differences between two files by displaying the first filename, followed by the lines that differ between the files, followed by the first line to match in both files. **Fc** then displays the name of the second file, followed by the lines that are different, followed by the first line that matches.

The default value for the number of lines to match between the files is 2. If you want to change this default, specify the number of lines with the `/nnnn` switch.

Limitations on comparisons

Fc uses a large amount of memory (enough to hold 100 lines) as buffer storage space to hold the text files. If these files are larger than available memory, **fc** compares what it can load into the buffer space. If it doesn't find a match in those portions of the files in the buffer space, **fc** stops and displays the following message:

```
resynch failed. Files are too different.
```

For binary files larger than available memory, **fc** compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as for those files that fit completely in memory.

Examples:

Suppose you want to compare two text files called *monthly.rpt* and *sales.rpt*. To make this comparison, you would simply type the following command line:

```
fc /a monthly.rpt sales.rpt
```

If you want to check for differences in files that are not ASCII text files, you can use the **/b** switch to force a binary comparison of the files. For example, if you have two executable program files called *profits.exe* and *earnings.exe*, and you want to find out whether they are the same, you could type the following:

```
fc /b profits.exe earnings.exe
```

The output from this command line might be similar to the following:

```
00000002: fc b6
00000004: 12 14
0000000e: 56 92
00000012: e8 5c
00000013: bb 7c
00000014: 14 0e
00000015: 0a 0d
0000001e: 43 7a
0000001f: 09 0a
00000022: be e6
...
...
...
000005e0: 00 61
000005e1: 00 73
000005e2: 00 73
000005e3: 00 69
000005e4: 00 67
000005e5: 00 6e
000005e6: 00 6d
000005e7: 00 65
000005e8: 00 6e
fc: earnings.exe longer than profits.exe
```

If the *profits.exe* and *earnings.exe* files were identical, **fc** would display the following message:

```
fc: no differences encountered
```

Comparing text files

Comparing binary files

Fdisk

Fdisk

Purpose:

Configures a hard disk for use with MS-DOS.

Syntax:

fdisk

Comments:

The **fdisk** command displays a series of menus to help you partition your hard disk for MS-DOS. With the **fdisk** command, you can

- Create a primary MS-DOS partition
- Create an extended MS-DOS partition
- Change the active partition
- Delete an MS-DOS partition
- Display partition data

Select the next fixed disk drive for partitioning on a system with multiple fixed disks

Notes:

- **Fdisk** doesn't work on drives used in the **subst** or **join** commands.
- For more information on how to use **fdisk**, see Appendix D, "Configuring Your Hard Disk (Fdisk)."

Find

Purpose:

Searches for a specific string of text in a file or files.

Syntax:

find [/v] [/c] [/n] "*string*" [[drive:][pathname] ...]

where:

string is a group of characters you want to search for.

Comments:

The **find** command looks for *string* in one or more files. After searching the specified files, **find** displays any lines it has found that contain the specified string.

String must be enclosed in quotation marks. Uppercase characters in *string* will not match lowercase characters.

If *string* contains quotation marks, you must enclose it in an additional pair (""*string*"").

If you omit *pathname*, **find** acts as a filter. It takes input from the MS-DOS standard input (usually from the keyboard, a pipe, or redirected file) and displays any lines that contain *string*.

Wildcards (* and ?) are not allowed in filenames or extensions.

The **find** command accepts the following switches:

Switch	Purpose
/v	Displays all lines <i>not</i> containing the specified string.
/c	Displays only the number of lines that contain a match in each of the files.
/n	Precedes each line with its relative line number in the file.

If you specify the /c switch with the /v switch, **find** displays the number of lines that do not contain the string you typed. If you specify the /c switch with the /n switch, **find** ignores the /n switch.

Find

The Find switches

Examples:

Finding a string of text in a file

The following command displays all lines from the file *pencil.ad* that contain the string "Pencil Sharpener":

```
find "Pencil Sharpener" pencil.ad
```

The next command causes MS-DOS to display the names of all files on the drive with the filename extension ".exe":

```
tree /f: | find ".exe"
```

In this last example, MS-DOS finds all the files with the filename extension *.exe* and lists them. Notice that the directory file created by the *tree* command is piped to the *find* command in order for *find* to perform the search.

If you want to find the string, "The dentist said, ""Open wide!""", in the file *story.doc*, type the following command:

```
find "The dentist said, ""Open wide!"""  
story.doc
```

Notice that in this last example, the entire phrase you were searching for was set off by quotation marks. To search for quotation marks in a string, use two sets of quotation marks.

Format

Purpose:

Formats the disk in the specified drive to accept MS-DOS files.

Syntax:

format drive:[/1]/4]/[8]/[n: sectors]/[t:tracks] [/v[:label]][/s]

or

format drive:[/1]/[b]/[n:sectors]/[t:tracks]

or

format drive:[/v[:label]]/[f: size]/[s]

Comments:

The **format** command creates the directory and the file allocation tables on a disk. **Format** also assigns a unique volume serial number to each disk. You must use this command to format all new disks before MS-DOS can use them.

Warning

Formatting destroys any previously existing data on a disk and it ignores drive assignments created with the **assign** command.

You must specify the drive that you want to use to format a disk.

Format then uses the drive type to determine the default format for a disk.

The **format** command accepts the following switches:

Switch	Purpose
/1	Formats a single-side of the floppy disk.
/4	Formats a 5.25-inch, 360K, double-sided disk in a high capacity disk drive. Some 360K drives may not be able to reliably read disks formatted with this switch.
/8	Formats eight sectors per track.
/b	Formats the disk, leaving ample space to copy an operating system, such as MS-DOS 4.0.
/s	Copies the operating system files listed in the file <i>formats.tbl</i> from the disk in the default drive to the newly formatted disk. The newly formatted disk must be 1.2 megabytes or greater in size; otherwise,

Format

The Format switches

Switch	Purpose
	format rejects the command. If the operating system is not on the default drive, format prompts you to insert a system disk in the default drive (or in drive A if the default drive is nonremovable).
/t:tracks	Specifies the number of tracks on the disk. This switch formats a 3.5-inch floppy disk to the number of tracks specified. For 720K-byte disks and 1.44-megabyte disks, this value is 80 (/t:80).
/n:sectors	Specifies the number of sectors per track. This switch formats a 3.5-inch disk to the number of sectors specified. For 720K-byte disks, this value is 9 (/n:9).
/v:label	Specifies the volume label to use. A volume label identifies the disk and can be up to 11 characters in length (no tabs allowed). If you do not specify the /v switch, or specify it without giving the volume label, you are automatically prompted for the volume label after the format operation completes.
/f:size	Specifies the size of the diskette to format. <i>Size</i> can be one of the following:

Disk Type	Size specifiers
160K single-sided 5.25-inch disk	160, 160K, 160KB
180K single-sided 5.25-inch disk	180, 180K, 180KB
320K double-sided 5.25-inch disk	320, 320K, 320KB
360K double-sided 3.5-inch disk	360, 360K, 360KB
720K double-sided 3.5-inch disk	720, 720K, 720KB
1.2M double-sided 5.25-inch disk	1200, 1200K, 1200KB, 1.2, 1.2M, 1.2MB
1.44M double-sided 3.5-inch disk	1440, 1440K, 1440KB, 1.44, 1.44M, 1.44MB

The **/n** and **/t** options cannot be used with **/f**.

When you format a hard disk, **format** prompts you to verify the volume label:

Formatting a hard disk

Enter current Volume Label for drive x:

If your hard disk does not have a volume label, press the **ENTER** key. But notice that if your hard disk has never been formatted before, or if it has a bad boot sector, **format** will not prompt you for a volume label.

If the volume label that you enter does not match the label on the hard disk, **format** displays the following message:

Invalid Volume ID Format failure

Otherwise it continues:

WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE X: WILL BE LOST!
Proceed with Format (Y/N)?_

If you want to format your hard disk, type *Y* (for Yes) and press the **ENTER** key. If you don't want to format your hard disk, type *N* (for No) and press the **ENTER** key.

When formatting is complete, **format** displays a message showing the total disk space, any space marked as defective, the total space used by the operating system (when you use the */s* switch), and the space available for your files.

Notes:

- You should not use the **format** command with drives used in the **assign**, **join**, or **subst** commands, and you cannot format drives over a network.
- For more information about formatting your hard disk, see Appendix D, "Configuring Your Hard Disk (Fdisk)."
- For more information about disk volume labels, see the **dir**, **label**, and **volume** commands in this chapter.

- The following table shows which switches you can use for certain types of disks:

Disk type	Valid switches
160/180 kilobytes	/1 /4 /8 /b /n /t /v /s /f
320/360 kilobytes	/1 /4 /8 /b /n /t /v /s /f
720 kilobytes	/b /n /t /v /s /f
1.2 megabytes	/b /n /t /v /s /f
1.44 megabytes	/b /n /t /v /s /f
hard disk	/b /v /s .

- The **format** command returns the following exit codes:

Code	Function
0	Successful completion
3	Terminated by user (CONTROL+C)
4	Fatal error (any error other than 0, 3, or 5)
5	N response to hard disk prompt, "Proceed with format (Y/N)?"

You can check these exit codes by using the *errorlevel* condition with the **if** batch processing command.

Examples:

Formatting a floppy disk

To format a floppy disk in drive A and copy the operating system to it, type the following command:

```
format a: /s
```

To format a floppy disk in drive A for use with data, type

```
format a: /v
```

Graftabl

Graftabl

Purpose:

Enables an extended character set to be displayed in graphics mode when using a color graphics adapter (CGA).

Syntax:

graftabl [xxx]

or

graftabl /sta[tus]

or

graftabl [?]

where:

xxx is a code page identification number.

Comments:

Valid code pages (xxx) include the following:

Value	Code Page
437	United States (default)
850	Multilingual
860	Portuguese
863	Canadian-French
865	Nordic

The **graftabl** command accepts the following switches:

Switch	Purpose
/sta[tus]	Displays the active character set.
?	Displays instructions for using the graftabl command.

The Graftable switches

If you type the **graftabl** command followed by the /sta[tus] switch, MS-DOS displays the active code page.

After **graftabl** loads the character table, it displays the following message:

Graphics characters loaded

You might want to put the **grftabl** command in your *autoexec.bat* file to save time. **Graftabl** may be used multiple times to change the loaded characters.

Notes:

- The **grftabl** command increases the size of MS-DOS resident in memory.
- For more information about using code pages, see the **chcp** command in this chapter.
- The **grftabl** command returns the following exit codes:

Code	Function
Graftabl exit codes	
0	Command successful. No previous code page loaded.
1	Table already loaded and replaced by new table.
2	File error occurred
3	Incorrect parameter, no action taken
4	Incorrect version of MS-DOS; version 4.0 required

You can check these exit codes using the *errorlevel* condition with the **if** batch processing command.

Examples:

Loading the graphics table

To load a table of graphics characters into memory, type

```
grftabl
```

Graphics

Graphics

Purpose:

Lets you print a graphics display screen on a printer when you are using a color or graphics monitor adapter.

Syntax:

graphics *type* [*profile*] [/r] [/b] [/lcd] [/printbox:*id*]

where:

profile is the name of the file that contains information on all supported printers. If this is omitted, the *graphics.pro* file is used.

type is one of the following:

Value	Function
COLOR1	Prints on an IBM Personal Computer Color Printer with black ribbon.
COLOR4	Prints on an IBM Personal Computer Color Printer with RGB (red, green, blue, and black) ribbon.
COLOR8	Prints on an IBM Personal Computer Color Printer with CMY (cyan, magenta, yellow, and black) ribbon.
GRAPHICS	Prints on an IBM Personal Graphics Printer, IBM Proprinter, or IBM Quietwriter Printer.
GRAPHICSWIDE	Prints on an IBM Personal Graphics Printer with an 11-inch wide carriage.
THERMAL	Prints on an IBM PC-convertible Thermal Printer.

Comments:

The **graphics** command supports the graphics display modes of the EGA, VGA, and 8514/A display adapters.

If you do not specify the *printer* option, **graphics** defaults to the GRAPHICS printer type.

The **graphics** command accepts the following switches:

The Graphics switches	Switch Purpose
/b	Prints the background in color. This option is valid for COLOR4 and COLOR8 printers.
/r	Prints black and white (as seen on the screen) on the printer. The default is to print black as white and white as black.
/lcd	Prints an image using LCD aspect ratio instead of CGA aspect ratio. This is the same as the /printbox:lcd switch.
/printbox:id	Selects the print box size. <i>id</i> should match the first operand of a printbox statement in the printer profile. The two options for this profile file are std or led . pb is the abbreviated form of printbox .

To print the screen, press the **SHIFT** and **PRINTSCREEN** keys at the same time. If the computer is in 320x200 color graphics mode, and if the printer type is COLOR1 or GRAPHICS, **graphics** prints the screen contents with up to four shades of gray. If the computer is in 640x200 color graphics mode, **graphics** prints the screen contents sideways on the paper.

Notes:

The **graphics** command increases the size of MS-DOS resident in memory.

Examples:

Printing a graphics screen

To print a graphics screen on your printer, type the following command:

```
graphics
```

Then, when the screen displays the information you want to print, press the **SHIFT** and **PRINTSCREEN** keys at the same time.

Join

Join

Purpose:

Joins a disk drive to a specific path.

Syntax:

join [*drive: drive:path*]

or

join *drive: /d*

Comments:

With the **join** command, you don't need to name physical drives with separate drive letters. Instead, you can create a directory name that will represent another drive. If the path already existed before you gave the **join** command, you cannot use it while the "join" is in effect. Also, you cannot join a drive if it is being used by another process.

If the *path* does not exist, MS-DOS tries to make a directory with that path. After you give the **join** command, the first drive name becomes invalid, and if you try to use it MS-DOS displays the "Invalid drive" error message.

If the directory to be joined is not empty, MS-DOS displays the "Directory not empty" error message.

Notes:

The following commands do not work on drives used in the **join** command (or the **subst** command):

backup	format
chkdsk	label
diskcomp	recover
diskcopy	restore
fdisk	sys

Examples:

Joining a drive

You can join a drive only with a root level directory. For example, this command will work:

```
join d: c:\sales
```

But the following one will *not*:

```
join d: c:\sales\regional
```

To reverse join, use the following format:

```
join drive:/d
```

Here, *drive*: represents the source drive, and the /d switch turns off the join command.

If you type just the join command by itself, MS-DOS displays the current drives that are joined.

Keyb

Keyb

Purpose:

Loads a keyboard program.

Syntax:

keyb [*xx*],[*yyy*],[*[drive:][path]filename*]]/[*ID:nnn*]

where:

xx is a two-letter country code.

yyy is the code page which defines the character set.

filename is the name of the keyboard definition file. If not specified, the filename used is *keyboard.sys*.

/ID:nnn specifies the keyboard in use.

Comments:

xx is one of the following two-letter, keyboard codes and *yyy* is one of the code pages:

Keyb Code	Country	Keyb ID	Code Page
be	Belgium	120	437,850
cf	Canadian-French	058	863,850
df	Denmark	159	865,850
fr/120	France	189/120	437,850
gr	Germany	129	437,850
it/142	Italy	141/142	437,850
la	Latin America	171	437,850
nl	Netherlands	143	437,850
no	Norway	155	865,850
po	Portugal	163	860,850
sf	Swiss-French	150	437,850
sg	Swiss-German	000	437,850
sp	Spain	172	437,850
su	Finland	153	437,850
sv	Sweden	153	437,850
uk/168	United Kingdom	166/168	437,850
us	United States (default)	103	437,850

Country codes

If you type **keyb** without options, MS-DOS displays a message like the following to show the current keyboard code and its related code page, and the current code page used by your console screen device (CON):

```
Current keyboard code: FR code page: 437
Current CON code page: 437
```

You can switch from the **keyb** program to the default (United States) keyboard format at any time by pressing **CONTROL+ALT+F1**. You can then return to the memory-resident keyboard program by pressing **CONTROL+ALT+F2**.

The **keyb** command lets you use characters that are not part of the normal (QWERTY) keyboard format. Using the **keyb** command with one of the two-letter codes above, you can type commands or text to MS-DOS using either the standard keyboard or a special keyboard.

Notice that the characters that appear on your screen when you type on a standard keyboard do not necessarily match the label on the key. For IBM PC and IBM AT-compatible keyboards, you can produce some characters in the non-United States keyboard sets by pressing **CONTROL+ALT** along with an appropriate character key. For IBM enhanced keyboards or compatibles, use **ALT+GR**. To produce accented (and unlabeled) characters, you press *dead* keys. Dead keys are keys that do not display a character when used alone, but when followed by a letter, display that letter with an accent.

Notes:

You can also include the appropriate **keyb** command in your *autoexec.bat* file so that you won't have to type it each time you start MS-DOS.

DOS 4.0 supports the following keyboards:

- PC1, PC XT
- PC AT
- IBM PC-convertible

You can enter **keyb** at the MS-DOS command line or you can include an **install** command in your *config.sys* file. The **keyb** command returns the following exit codes:

Code	Function
0	Successful completion
1	Invalid syntax
2	Bad or missing keyboard definition file
3	Could not create keyboard table in resident memory
4	Error with CON device
5	Code page requested not prepared
6	Table for selected code page cannot be found in the resident keyboard table
7	Incorrect MS-DOS version

Examples:

To use a German keyboard, type the following command:

```
keyb gr
```

Appendix E, "How to Use Code Pages", provides additional examples of the **keyb** command.

Changing keyboard programs

Label

Label**Purpose:**

Creates, changes, or deletes the volume label on a disk.

Syntax:

label [*drive:*][*label*]

where:

label is the new volume label, up to 11 characters.

Comments:

A *volume label* is a name you can specify for a disk. MS-DOS displays the volume label of a disk as a part of its directory listing to show you which disk you are using.

If a *volume serial number* exists, **label** will also display this eight-character number:

```
Volume Serial Number in drive x is nnnn-nnnn
```

If you do not specify a label, **label** prompts you with the following message:

```
Volume in drive x is xxxxxxxxxxxx
Volume Serial Number is xxxx-xxxx
Volume label (11 characters, ENTER none)?
```

Naming volume labels

A volume label may be up to 11 characters in length and may include spaces, but not tabs. Type the volume label that you want and press the ENTER key. Or, you can press the ENTER key immediately if you want to delete the volume label. **Label** will prompt you with the message:

```
Delete current volume label (Y/N)? _
```

If you type *Y* (for yes), **label** deletes the volume label on the disk. Otherwise, the volume label stays the same.

Notes:

- You can use the MS-DOS **dir** or **vol** command to determine if the disk already has a volume label.
- **Label** doesn't work on drives involved with **subst** or **join** commands.
- Do not use any of the following characters in a volume label:
* ? / \ | . , ; : + = < > [] () & ^

Examples:

To label a disk in drive A that contains sales information for 1988, you might type

```
label a:sale1988
```

Labeling a disk

Mem

Mem**Purpose:**

Displays the amount of used and free memory, lists allocated and free memory areas, and programs that are loaded.

Syntax:

mem [/program]

or

mem [/debug]

Comments:

Mem accepts the following switches:

Switch	Purpose
/program	Displays programs loaded in memory.
/debug	Displays programs, internal drivers, and other programming information.

Notes:

You cannot specify **/program** and **/debug** at the same time.

Extended memory is displayed only if memory above 1M is installed in the system. Expanded memory is displayed only if a LIM 4.0 driver is installed. All DOS 4.0 components that allocate Expanded Memory Specification (EMS) memory define a name for the EMS handle returned. This name is set to the component name (for example, **buffers**, **fastopen**, or **RamDrive**) which is used in the **/debug** switch display to identify EMS memory usage.

Examples:

To display memory information for a 640K system that uses expanded memory, type the following:

```
mem /program
```

The Mem switches

The output might look like this:

Address	Name	Size	Type
DDDDDDD	DDDDDDD	DDDDDD	DDDDDD
000000		000400	Interrupt Vector
000400		000100	ROM Communication Area
000500		000200	DOS Communication Area
000700	IO	002310	System Program
002A10	MSDOS	008F10	System Program
00B920	IO	0069B0	System Data
	DISPLAY	0037A0	DEVICE=
	DRIVER	0001F0	DEVICE=
		0000D0	FILES=
		000110	FCBS=
		001F40	BUFFERS=
		0001C0	LASTDRIVE=
		000CD0	STACKS=
0122E0	COMMAND	001810	Program
013B00	FASTOPEN	000030	Environment
013B40	COMMAND	0000A0	Environment
013BF0	RETRIEVE	000040	Environment
013C40	RETRIEVE	003280	Program
016ED0	FASTOPEN	0020F0	Program
018FD0	MEM	000030	Environment
019010	MEM	012620	Program
02B640	MSDOS	0749B0	- Free -

655360 bytes total memory
 655360 bytes available
 552928 largest executable program size

1703936 bytes total EMS memory
 1015808 bytes free EMS memory

524288 bytes total extended memory

Mkdir (md)

Mkdir**Synonym:**

md

Purpose:

Makes a directory.

Syntax:

mkdir [*drive:*]*path*

or

md [*drive:*]*path*

Comments:

The **mkdir** command lets you create a multilevel directory structure. The maximum length of any single path from the root directory to the desired level cannot be more than 63 characters, including embedded backslashes.

Examples:

Creating a subdirectory

If you want to create a directory to keep all your tax information, you could type the following command from your root directory:

```
mkdir \taxes
```

Now, suppose you want to create a directory named *rental* under the *\taxes* directory to keep track of information about a duplex that you rent out. To do this from the root directory, you simply type the following command:

```
mkdir \taxes\rental
```

To create the same subdirectory from the *\taxes* directory, you could type either the previous command, or

```
mkdir rental
```

Mode

Mode

Purpose:

The **mode** command has several uses:

- To display the status of all devices or of a single device.
- To reconfigure a printer attached to a parallel port (PRN, LPT1, LPT2, or LPT3) for printing at 80 or 132 characters per line, 6 or 8 lines per inch, or both (if the printer supports these features). In this form, **mode** can also be used to select a parallel printer other than the one attached to LPT1 for use as the default printer.
- To select another display or to change the configuration of the current display. Changing the configuration includes changing between a 40-column and an 80-column display, changing between monochrome and color display, changing the number of lines on the display, centering the display on a screen, or any combination of these.
- To configure the baud rate, parity, and number of databits and stopbits of a serial communications port (COM1, COM2, COM3, and COM4) for use with a specific printer, modem, or other serial device.
- To set the keyboard's typematic rate.
- To prepare devices for code-page switching.
- To redirect printer output for a parallel port to one of the serial ports, so that the serial port becomes the system's default printer port.

Because the syntax for each of these uses of the **mode** command is different, they are discussed separately on the following pages.

Note

Some functions, like setting the keyboard or display mode, can be performed only if the *ansi.sys* driver is installed in your *config.sys* file.

Although each form of the **mode** command can be issued at the system prompt, **mode** commands are commonly used within the file to automatically perform any necessary reconfiguration each time the system is turned on or restarted. See Chapter 4, "Batch Processing", for more information on the *autoexec.bat* file.

Mode - Device status

Purpose:

Display device status.

Syntax:

mode[*device*][/**sta**[**tus**]

Comments:

Typing **mode** by itself displays the status of all devices installed in your system. The **/sta[tus]** switch is required only when requesting status for (redirected) parallel printers.

Example:

To display the status of a specific device, type **mode** followed by the device name. For example, type:

mode con

Mode - Configure printer

Purpose:

Sets the characteristics for IBM-compatible printers connected to a parallel printer port (LPT1, LPT2, or LPT3).

Syntax:

mode LPT *n*[:*c*][:*l*][:*r*]

or

mode LPT*n*[*cols=c*][*lines=l*][*retry=r*]

For parallel printer modes, you can use PRN and LPT1 interchangeably. You can use the following options with the **mode** command to set parameters for a parallel printer:

Parallel printer modes

Option	Purpose
<i>n</i>	Specifies the printer number: 1, 2, or 3.
<i>c</i>	Specifies the characters per line: 80 or 132.
<i>l</i>	Specifies the vertical spacing and lines per inch: 6 or 8.
<i>r</i>	Specifies the retry action to take when time-out errors occur. This option causes part of the mode program to remain resident in memory. If you are using the mode command over a network, do not use the <i>r</i> option for continuous retry. Possible values are as follows: E returns the error from a status check of a busy port (default). B returns busy from a status check of a busy port. R returns busy from a status check of a busy port. NONE means no retry action is taken.

In the **mode** command, the **retry=B** option provides the same support as the **p** parameter in previous releases.

The default settings are LPT1, 80 characters per line, and 6 lines per inch.

You can break out of a time-out loop by pressing **CONTROL+BREAK**.

Examples:

Sending output to a parallel printer

Suppose you want your computer to print on a parallel printer that is connected to your computer's second parallel printer port (LPT2). If you want to print with 80 characters per line and 8 lines per inch, you would type

```
mode lpt2: 80,8
```

or

```
mode lpt2: ,8
```

If you want your computer to keep trying to print a file until your printer is ready to print it, type

```
mode lpt2:80,8,B
```

To stop retrying to print, you can press **CONTROL+BREAK** or type the **mode** command without the **r** option.

.

Mode - Configure serial port

Purpose:

Controls the configuration of the serial communications adapter.

Syntax:

```
mode COMm[:]b,[p],[d],[r]]]
```

or

```
mode COMm baud=b[data=d] [stop=s][parity=p][retry=r]
```

Comments:

You can use the following options with the **mode** command to set parameters for serial ports:

Option	Purpose
<i>m</i>	Specifies the asynchronous communications (COM) port number: 1, 2, 3, or 4.
<i>b</i>	Specifies the first two digits of the transmission rate: 110, 150, 300, 600, 1200, 2400, 4800, 9600, or 19,200.
<i>p</i> .	Specifies the parity: N (none), O (odd), E (even), M (mark), or S (space). The default value is E.
<i>d</i>	Specifies the number of data bits: 5, 6, 7, or 8.
<i>s</i>	Specifies the number of stop bits: 1, 1.5, or 2. If baud rate is 110, the default value is 2; otherwise, the default value is 1.
<i>r</i>	Specifies what type of retry action to take. If you are using the mode command over a network, do not use the r option for continuous retry. Options are E returns an error from a status check of the busy port (default). B returns busy from a status check of the busy port. R returns ready from a status check of the busy port. NONE means that no retry action is taken.

In the **mode** command, the **retry=B** option provides the same support as the **p** parameter in previous releases.

The default settings are COM1, even parity, and 7 databits.

Mode - Set display mode

Purpose:

Selects the active video adapter and its display mode or reconfigures the current display.

Syntax:

mode *display*, *n*

or

mode [*display*],*shift*[,*test*]

or

mode con [:] [*cols*=*m*][*lines*=*n*]

Comments:

You can use the following options with the **mode** command to set parameters for a display:

Option	Purpose
<i>display</i>	Specifies one of the following values: 40, 80, BW40, BW80, CO40, CO80, or MONO. 40 and 80 indicate the number of characters per line. BW and CO refer to a color graphics monitor adapter with color disabled (BW) or enabled (CO). MONO specifies a monochrome display adapter with a constant display width of 80 characters per line.
<i>shift</i>	Specifies whether to shift a Color Graphics Adapter (CGA) display to the left or to the right. Valid values are L (for left) or r (for right).
<i>n</i>	Specifies the number of lines on the display. Possible values are 25, 43, and 50. Not all adapters support all sizes.
<i>m</i>	Specifies the number of characters per line. Possible values are 40 or 80.
<i>test</i>	Allows you to align the display. You are prompted to indicate whether or not the screen is correctly aligned.

Mode - Set keyboard mode

Purpose:

Set keyboard typematic rate.

Syntax:

mode con [:] rate=*r* delay=*d*

Comments:

To set the typematic rate, you must specify both the rate and delay at the same time.

Option	Purpose
<i>r</i>	Specifies the typematic interval time. The range for possible values is 1-32.
<i>d</i>	Specifies the auto-repeat start delay time. Possible values are 1-4 (effective values are .25, .50, .75, and 1 second).

Mode - Set device code pages

Purpose:

Set or display code pages for parallel printers or your console.

Syntax:

mode device codepage prepare=(*yyy*)[*drive:*][*path*]*filename*)

and

mode device codepage select=*yyy*

mode device codepage refresh

mode device codepage [*/status*]

Comments:

You can use the following options with **mode** to set or display code pages:

Option	Purpose
<i>device</i>	Specifies the device to support code-page switching. Valid <i>device</i> names are CON, LPT1, LPT2, and LPT3.
<i>yyy</i>	Specifies a code page. Valid code pages are 437, 850, 860, 863, and 865.
<i>filename</i>	Identifies the name of the Code Page Information (.cpi) file MS-DOS should use to prepare a code page for the device specified.

There are four keywords that you can use with the **mode device codepage** command. Each causes the **mode** command to perform a different function. The following table explains each keyword:

Keyword	Function
prepare	Tells MS-DOS to prepare code pages for a given device. You must prepare a code page for a device before you can use it with that device.
select	Specifies which code page you want to use with a device. You must prepare a code page before you can select it.

Keyword	Function
---------	----------

refresh	If the prepared code pages for a device are lost due to a hardware or other error, this keyword reinstates the prepared code pages.
----------------	---

/status	Displays the current code pages prepared and/or selected for a device. Notice that the following commands both produce the same results:
----------------	--

```
mode con codepage
```

```
mode con codepage /status
```

Typing "/status" is optional.

Notes:

- You can use the following abbreviations with the **mode** command for code page modes:

Type	In place of
cp	codepage
/sta	/status
prep	prepare
sel	select
ref	refresh

For more information about using the **mode** command to set or display code pages, see Appendix E, "How to Use Code Pages".

Mode - Redirect printing

Purpose:

Redirects output from a parallel port to a serial communications port.

Syntax:

mode LPTn[:]=COM n[:]

Examples:

Redirecting output

Suppose you want your computer to send its printer output to a serial printer. To do this, you need to use the **mode** command twice. The first **mode** command configures the serial port, and the second **mode** command redirects the computer's parallel printer output to the serial port specified in the first **mode** command.

For example, if your serial printer operates at 4800 baud with even parity, and if it is connected to the COM1 port (the first serial connection on your computer), you would type **m**

```
ode com1:48,e,,B
```

```
mode lpt1:=com1:
```

If you have redirected the parallel printer output from LPT1 to COM1, and then decide that you want to print a file using LPT1, type

```
mode lpt1:
```

This command disables any redirection of LPT1.

More

More

Purpose:

Displays output one screen at a time.

Syntax:

more < *source*

or

source | **more**

where:

source is a file or command.

Comments:

More is a filter that reads from standard input (from a pipe or redirected file) and displays one screen of information at a time. **More** is commonly used to view long files.

For example, you may use the **dir** command, the **sort** command, or a filename as a source. The **more** command then pauses and displays the *--More--* message at the bottom of your screen.

Press any key to display another screen of information. Then, keep pressing the key until you have read all the data.

Notes:

To hold input information until it is displayed, the **more** command creates a temporary file on the disk. If the disk is full or write-protected, **more** will not work.

See Also:

For more information about using redirection symbols with commands, see Chapter 2, "Using MS-DOS Commands".

Examples:

Suppose you have a long file called *clients.new* that you want to view on your screen. The following command redirects the file through the **more** command to show the file's contents one screen at a time:

```
more < clients.new  
or  
type clients.new | more
```

Displaying output screen-by-screen

Viewing long files

Nlsfunc

Nlsfunc**Purpose:**

Loads country-specific information.

Syntax:

```
nlsfunc[[drive:][path]filename]
```

where:

filename specifies the file containing country-specific information.

Comments:

The **nlsfunc** command supports the use of extended country-specific information and code page switching.

The default value of *filename* is defined by the **country** command in your *config.sys* file. If no **country** command exists in your *config.sys* file, MS-DOS uses the *country.sys* file in your root directory for country-specific information.

Notes:

You can use an **install** command in your *config.sys* file to install *nlsfunc.exe*.

For more information about using the **nlsfunc** command, see Appendix E, "How to Use Code Pages".

Examples:

Suppose you have a file on your disk called *newcdpg.sys* that contains country-specific information. If you want to use the information from that file rather than the *country.sys* file, you would type the following command:

```
nlsfunc newcdpg.sys
```

To use the default country-specific information found in the *country.sys* file, simply type this command:

```
nlsfunc
```

**Loading country-specific
Information**

Path

Purpose:

Sets a command search path.

Syntax:

path [*drive:*][*path*][*;*][*drive:*][*path*]...

or

path ;

Comments:

The maximum length of a command search path is 127 characters. The **path** command lets you tell MS-DOS which directories to search for executable commands—after it searches your working directory. Executable commands include files with a filename extension of *.com*, *.exe*, or *.bat*. The default value is no path.

For instance, to tell MS-DOS to search the *c:\user\pete* directory for executable commands, you would simply type the **path** command followed by the directory name *\user\pete*. Then, until you exit MS-DOS or set another path, MS-DOS searches the *c:\user\pete* directory for executable commands. The commands are returned in the following order of precedence: *.com*, *.exe*, and *.bat*.

You can tell MS-DOS to search more than one path by specifying several paths separated by semicolons. If you use the **path** command without options, it prints the current path. And if you use the following command, MS-DOS searches only the working directory for executable commands:

```
path ;
```

This path is also called the NUL path.

Examples:

The following command tells MS-DOS to search three directories to find external commands (the three paths for these directories are *c:\user\pete*, *b:\user\emily*, and *\bin*):

```
path c:\user\pete;b:\user\emily;\bin
```

MS-DOS searches the paths in the order specified in the **path** command.

Path

Setting a search path

Print**Print (PRN)****Purpose:**

Prints a text file on a lineprinter *while you are processing* other MS-DOS commands (usually called *background printing*).

Syntax:

```
print [/d:device][/b:size][/u:value1] [/m:value2]
[/s:timeslice][/q:qsize][/t][drive:][pathname][c][p]
```

Comments:

You can use the **print** command only if you have an output device, such as a printer or a plotter, connected to one of your computer's serial or parallel ports.

The following are the switches accepted by the **print** command:

The Print switches

Switch	Purpose
/d:device	Specifies the print device name. If you specify /d , it must be the first switch. The default device is PRN. Other possible print device names for parallel ports are LPT1, LPT2, and LPT3. COMx, where x is a number from 1 to 4, refers to a serial port. (LPT1 and PRN both refer to the first parallel port on your computer.)
/b:size	Sets the size in bytes of the internal buffer. To speed up the print command, you increase the value of /b . The minimum and default value of /b is 512, the maximum is 1634.
/u:value1	Specifies the number of clock ticks print will wait for a printer. If the printer is not available within the time specified, the job will not run. Valid values for <i>value1</i> range from 1 to 255. The default for <i>value1</i> is 1.
/m:value2	Specifies the number of clock ticks print can take to print a character on the printer. Valid values for <i>value2</i> range from 1 to 255. The default is 2.
/s:timeslice	The interval of time to be used by the MS-DOS scheduler for the print command. Valid values for <i>timeslice</i> range from 1 to 255. The default is 8.

Switch	Purpose
/q:qsize	Specifies the number of files allowed in the print queue. The minimum value for the /q switch is 4, the maximum, 32, and the default, 10. To change this default number of files, you must use the print command without any filenames; for example, print /q:32
/t	Deletes all files in the print queue (those files waiting to be printed).
/c	Turns on the cancel mode and removes the preceding filename and all following filenames from the print queue. You must specify a filename before the /c switch.
/p	Turns on the print mode and adds the preceding filename and all following filenames to the print queue. You must specify a filename before the /p switch.

The **print** command, when used with no options, displays the contents of the print queue on your screen without affecting the queue.

Notes:

- The /d, /b, /u, /m, /s, and /q switches are allowed only the first time you run the **print** command after starting MS-DOS.
- Each print queue entry may contain a maximum of 64 characters, including the drive name. So you may need to change directories first to avoid using extensive pathnames.
- Some applications have their own print commands. You should use the application's print facility to print files that you create with the application.

Examples:

The following command sets up the print queue for printing on LPT1:

```
print /d:lpt1
```

The following command removes the *pencil.tst* file from the default print queue:

```
print a:pencil.tst /c
```

The next two commands show how to remove the file *pencil.tst* from the queue and then add the file *pen.tst* to the queue:

```
print pencil.tst /c  
print pen.tst /p
```

Prompt

Purpose:

Changes the MS-DOS command prompt.

Syntax:

prompt *[[text][\$character]...]*

Comments:

This command lets you change the MS-DOS system prompt (for example, A>). If, when using the **prompt** command, you do not type a new value, the prompt is set to the default value, which includes the default drive name.

You can use the characters in the **prompt** command to create special prompts:

Type these characters	To get this prompt
\$q	The = character
\$\$	The \$ character
\$t	The current time
\$d	The current date
\$p	The working directory of the default drive
\$v	The version number
\$n	The default drive
\$g	The >character
\$l	The <character
\$b	The character
\$_	ENTER+LINEFEED
\$e	ASCII code X'1B' (escape)
\$h	Backspace (to erase a character that has been written to the prompt line)

Examples:

The following example sets the drive prompt to *drive: current directory*:

```
prompt $p
```

Prompt

Special prompts

Setting your prompt

The following command sets a two-line prompt that displays the following:

```
Time = (current time)
Date = (current date)

prompt time = %t%_date = %d
```

**Using an ANSI driver
to set inverse video**

If your terminal has an ANSI escape sequence driver, you can use escape sequences in your prompts. The following command, for example, sets your prompt in inverse video mode and returns to video mode for other text:

```
prompt $e[7m$n:$e[m
```

Note

For information on *ansi.sys*, refer to Appendix C, "Installable Device Drivers".

Recover

Recover

Purpose:

Recovers readable information from a file or files when one or more sectors on the disk become bad or defective.

Warning

Since the root directory can only hold a finite number of entries, some of the recovered files may be lost. If all files on a disk need to be recovered, recover the files one at a time. Do not attempt to recover the files by subdirectory or by recovering the entire disk at one time.

Specifying a subdirectory in the `recover` command line causes the subdirectory to be recovered to the root directory, rather than to the original subdirectory where the files were stored.

Specifying the entire disk (or drive) causes all files on the disk to be recovered on the root directory. Each recovered file is placed on the root directory in a `filennnn.rec` file, where `nnnn` is a sequential number starting with 0001. Use this form of the `recover` command if the directory of the disk is not usable.

Since all data in the bad sectors are lost after recovery is run on a file, recovering files singly allows you to edit the files once they are recovered and to re-enter any missing information.

Syntax:

`recover [drive:][path]filename`

If recovering a single file.

or

`recover drive`

The second syntax should only be used if all files on the current drive need to be recovered.

Comments:

If the `chkdsk` command shows that a sector on your disk is bad, you can use the `recover` command to recover each file on the disk or just the file containing the bad sector.

The `recover` command causes MS-DOS to read the file sector by sector and to skip the bad sectors. When MS-DOS finds a bad sector, it no longer allocates your data to that sector.

Notes:

- The **recover** command does not work on a network from a remote workstation.
- **Recover** does not work on drives used in the **subst** or **join** commands.
- The **recover** command does not work with the **backup** or **restore** commands. You must use the **restore** command with files backed up with the **backup** command.

Examples:

Recovering a file

Suppose you have a file named *pencil.ad* that has a few bad sectors. To recover this file, you would use the following command:

```
recover pencil.ad
```

If the disk or directory of drive A is unusable, enter the following command to recover all the files on the disk:

```
recover a:
```

Ren (Rename)

Ren (Rename)

Purpose:

Changes the name of a file.

Syntax:

```
rename [drive:][path]filename1 filename2
```

or

```
ren [drive:][path]filename1 filename2
```

where:

filename1 is the old name.

filename2 is the new name

Comments:

The **ren** command renames all files matching *filename1*. However, because you cannot rename files across disk drives, the **ren** command ignores any drive name that you specify with *filename2*.

You may use wildcards (* or ?) in either filename option, but if you use them in *filename2*, **ren** will not change the positions of the corresponding character.

Examples:

The following command changes the extension of all filenames ending in *.txt* to *.doc*:

```
ren *.txt *.doc
```

In the next example, **ren** renames a file named *chap10* (on drive B) to *part10*:

```
ren b:chap10 part10
```

The newly renamed file *part10* remains on drive B.

Renaming files

Replace

Replace**Purpose:**

Updates previous versions of files.

Syntax:

replace [*drive:*]*pathname1* [*drive:*][*pathname2*] [/a]/p[/r]/s[/w]/u]

where:

pathname1 is the source path and filename.

pathname2 is the target path and filename.

Comments:

The **replace** command performs two functions:

- It replaces files in the target directory with files in the source directory that have the same name.
- When you specify the /a switch, **replace** adds files that exist in the source directory (but *not* in the target directory) to the target directory.

You may use wildcards in source filenames.

The **replace** command accepts the following switches:

Switch	Purpose
/a	Adds new files to the target directory instead of replacing existing ones. You may not use this switch with the /s or /u switch.
/p	Prompts you with the following message before it replaces a target file or adds a source file: Replace <i>filename</i> ? (Y/N)_
/r	Replaces read-only files as well as unprotected files. If you do not specify this switch, any attempt to replace a read-only file causes an error and stops the replace process.
/s	Searches all subdirectories of the target directory while it replaces matching files. This switch is incompatible with the /a switch. Replace never searches subdirectories in the source path.

The Replace switches

Switch	Purpose
/u	Replaces only those files on the target directory that are older than those in the source directory. This switch compares the time/date stamp on the source files with that of the target files and replaces only those files that need updating.
/w	Waits for you to insert a disk before beginning to search for source files. If you do not specify the /w switch, replace begins replacing or adding files immediately.

If you specify **/w**, **replace** displays the following message:

```
Press any key to continue
```

As files are replaced or added, **replace** displays their filenames on the screen. At the conclusion of the **replace** operation, it displays a summary line:

```
nnn file(s) added/replaced
```

```
or
```

```
No files added/replaced
```

Notes:

- You cannot use the **replace** command to update hidden files or system files (*io.sys* and *msdos.sys*).
- Upon completion, **replace** returns one of the following exit codes:

Code	Function
0	Command successful
2	File not found
3	Path not found
5	Access denied
8	Insufficient memory
11	Command line error
15	Invalid drive

Replace exit codes

You can test for these codes by using the *errorlevel* condition of the batch processing **if** command.

Examples:

Replacing files

Suppose various directories on your hard disk, drive C, contain files named *phones.cli* that contain client names and phone numbers. To update these files and replace them with the latest version of the *phones.cli* file on the disk in drive A, you would type the following command:

```
replace a:\phones.cli c:\ /s
```

This command replaces every file on drive C that is named *phones.cli* with the file *phones.cli* from the root directory on drive A.

Adding files

Suppose you want to add some new printer device drivers to a directory called *c:\mstools*, which already contains several printer driver files for a word processor. To do this, you would type the following:

```
replace a:*.prd c:\mstools /a
```

This command searches the default directory of drive A for any files that have the extension *.prd* (that don't currently exist in the *\mstools* directory on drive C) and then adds these files to *c:\mstools*.

Restore

Restore

Purpose:

Restores files that were backed up using the **backup** program.

Syntax:

```
restore drive1: [drive2:][pathname][/s][/p][/b:date]  
[/a:date][/e:time][/L:time][/m] [/n]
```

where:

drive1 contains the backed-up files.

drive2 is the target drive.

pathname identifies the file(s) you want to restore.

Comments:

The **restore** command can restore files from similar or dissimilar disk types.

The **restore** command accepts the following switches:

Restoring to different media

Switch	Purpose
/s	Restores subdirectories also.
/p	Prompts for permission to restore any files matching the file specification that are read-only or that have changed since the last backup.
/b:date	Restores only those files last modified on or before <i>date</i> .
/a:date	Restores only those files last modified on or after <i>date</i> .
/e:time	Restores only those files last modified at or earlier than <i>time</i> .
/L:time	Restores only those files last modified at or later than <i>time</i> .
/m	Restores only those files modified since the last backup.
/n	Restores only those files that no longer exist on the target disk.

The Restore switches

Once MS-DOS has restored the file, use the **dir** or **type** command to make sure that the file was restored properly.

Notes:

- Restore cannot restore the system files.
- The MS-DOS 4.0 **restore** command will restore files backed up with (either the MS-DOS 4.0 **backup** command or an earlier version of) **backup**.
- Upon completion, **restore** returns one of the following exit codes:

Restore exit codes	Code	Function
	0	Normal completion
	1	No files were found to restore
	3	Terminated by user
	4	Terminated due to error

You can test for these codes by using the *errorlevel* condition of the **if** batch processing command.

Examples:

Restoring a file

To restore the file *invest.mnt* from the backup disk in drive A to the *\irsharpe* directory on drive C, type the following:

```
restore a: c:\irsharpe\invest.mnt
```

Press the **ENTER** key to let MS-DOS know that the backup disk is in drive A. Then once MS-DOS has restored the file, use the **dir** or **type** command to make sure that the file was restored properly.

Rmdir (rd)

Rmdir (rd)

Purpose:

Removes a directory from a multilevel directory structure.

Syntax:

rmdir [*drive:*]*path*

or

rd [*drive:*]*path*

Comments:

Rmdir removes a directory that is empty except for the "." and ".." symbols. These two symbols refer to the directory itself and its parent directory, respectively. Before you can remove a directory entirely, you must delete its files and subdirectories:

Notes:

You cannot remove a directory that contains hidden files. In addition to MS-DOS, which has some hidden files, certain application programs also create their own hidden files.

Examples:

Suppose you want to remove a directory named *\user\pete*. You would follow these steps:

Removing a directory

- 1 To ensure that the directory is empty, type the following:

```
dir \user\pete
```

- 2 Then, from any directory except the one you want to remove, type the following command:

```
rmdir \user\pete
```

Remember that if you are working in the same directory that you are trying to remove, you'll receive the following error message:

```
Attempt to remove current directory-.
```

Select

Select**Purpose:**

Installs MS-DOS on a hard disk or on floppy disks and creates *autoexec.bat* and *config.sys* files containing the desired configuration information.

Syntax:

With a version of MS-DOS earlier than 4.0 on your system, insert the Install disk in drive A, and then press **CONTROL+ALT+DELETE** to initialize the select program.

If you have already installed MS-DOS on the hard disk and you need to update the specifications made previously, insert the Install disk in drive A, and enter the following at the DOS prompt:

select menu

The Install disk must be in drive A before you can make any changes to the system specifications. Once the desired changes are made, **select** allows you to install the updated files on the hard disk. New versions of the *config.sys* and *autoexec.bat* files are copied over to the root directory.

Note

If *config.sys* and *autoexec.bat* do not exist, they are created. If *config.sys* and *autoexec.bat* do exist, the new versions of the files are named *config.400* and *autoexec.400*. For the new configuration to take effect, you will need to update your current *config.sys* and *autoexec.bat* with the information contained in *config.400* and *autoexec.400* and press **CONTROL+ALT+DELETE** to restart your system.

Comments:

The **select** command lets you install MS-DOS and perform the following functions:

- Specify how the available system memory will be shared between MS-DOS and application programs.
- Specify the country and keyboard (this determines date and time format, and currency symbols).

- Specify printer type and printer port (you may also designate a serial port as a parallel port if necessary).
- Specify various installation options including: code-page switching, expanded memory support, extended display support (*ansi.sys*), file performance enhancements (*fastopen*), *graftab* display support, *graphics* Prt Sc support, DOS share support, DOS shell support, and ram disk support (*ramdrive.sys*).
- Create *config.sys* and *autoexec.bat* files containing the specifications made while running *select*.

Notes:

- The default for each entry screen of the *select* program is highlighted. Use the **UP** and **DOWN** keys to choose an item other than the screen default selection. In addition, each entry screen has extensive help information that can be displayed by pressing the **F1** key. Refer to these files for information that will help you to make the selection that is right for your system.
- Before you begin *select*, make certain you know the following: how much memory is available with your current system, what type of printer you have (serial or parallel), and whether or not your system includes an expanded memory card. Having this ready will allow you to respond more quickly to many of the prompts displayed during the *select* program.

Set

Set**Purpose:**

Sets one string of characters in the environment equal to another string for later use in programs.

Syntax:

```
set [string=[string]]
```

Comments:

When MS-DOS recognizes a **set** command, it inserts the given *string* and its equivalent into a part of memory reserved for the *environment*. If the *string* already exists in the environment, it is replaced with the new setting.

If you specify just the first *string*, **set** removes any previous setting of that *string* from the environment. Or if you use the **set** command without options, MS-DOS displays the current environment settings.

When batch processing, you can also use the **set** command to define your replaceable parameters by name instead of by number. For example, if your batch file contains the statement *type %file%*, you could use the **set** command to set the name that MS-DOS will use for that variable. In the following command, for example, **set** stores a value for MS-DOS to replace the *%file%* parameter with the filename *taxes.86*:

```
set file=taxes.86
```

To change the replaceable parameter names, you don't need to edit each batch file. Notice also that when you use text (instead of a number) as a replaceable parameter, the name must be ended by a percent sign.

The **set** command is especially useful in the *autoexec.bat* file, because it lets you automatically set strings or parameters when you start MS-DOS. See Chapter 4, "Batch Processing", for more information about the *autoexec.bat* file.

After entering the **set** command, you might get the message "Out of environment space". If you do, refer to Appendix B, "How to Configure Your System, for information on increasing the environment space.

Defining replaceable parameters

Examples:

The following command sets the string "include" to *c:\inc* until you change it with another set command:

```
set include=c:\inc
```

Setting a string

Share

Share**Purpose:**

Installs file sharing and locking.

Syntax:

share [/f:space][/L:locks]

Comments:

Share is normally used in a network environment where files will be shared between processes. If **share** is installed, all read and write requests of an application are validated against the file sharing code in which the file was opened.

If **share** is installed, FCBS control is activated. Refer to Appendix B, "How to Configure Your System", for details on the FCBS command.

If a read or write operation takes place when a disk has been changed in the middle of the operation, **share** looks at the volume label to determine if it is the same disk that was previously in the drive. If it is not, MS-DOS sends an invalid disk error message and requests that you insert the proper disk in the drive. When the proper disk is inserted, **share** allows the *read/write* operations to be completed successfully.

Share is automatically loaded if you have a hard disk which is greater than 32 megabytes in a single partition.

Share can be executed by using the **install** command. Refer to Appendix B, "How to Configure Your System", for details on the **install** command.

To learn more about sharing and locking files, see the *Microsoft MS-DOS Programmer's Reference*.

The **share** command accepts the following switches:

Switch	Purpose
--------	---------

/f:space	Allocates file space (in bytes) for the MS-DOS storage area used to record file sharing information. The default value for the /f switch is 2048. Notice that each open file requires enough space for the length of the full filename plus 11 bytes, since an average pathname is 20 bytes in length.
/L:locks	Allocates the number of locks you want to allow. The default value for the /L switch is 20.

The Share switches

Once you have used the `share` command in an MS-DOS session, all read and write requests are checked by MS-DOS.

Example:

The following example loads file sharing and uses the default values for the `/f` and `/L` switches:

```
share
```

**Loading the Share
command**

Sort

Sort**Purpose:**

Reads input, sorts the data, then writes the sorted data to your screen, to a file, or to another device.

Syntax:

```
[source] | sort [/r][/+n]
```

or

```
sort [/r][/+n] < source
```

where:

source is a filename or command.

Comments:

The **sort** command is a filter program that lets you alphabetize a file according to the character in a certain column. The **sort** program uses the collating sequence table, based on the country code and code page settings.

The **|** and **<** redirection symbols direct data through the **sort** utility from *source*. For example, you may use the **dir** command or a filename as a source. You may use the **more** command or a filename as a destination.

The **sort** command accepts the following switches:

Switch	Purpose
/r	Reverses the sort; that is, sorts from Z to A, and then from 9 to 0.
/+n	Sorts the file according to the character in column <i>n</i> , where <i>n</i> is some number. If you do not specify this switch, the sort command sorts the file according to the character in the first column.

The Sort switches

Unless you specify a source, **sort** acts as a filter and accepts input from the MS-DOS standard input (usually from the keyboard, from a pipe, or redirected from a file).

Notes:

- Sort does not distinguish between uppercase and lowercase letters.
- Characters above ASCII code 127 are sorted based on information found in the *country.sys* file, or in an alternate file specified by the **country** command in your *config.sys* file.
- For more information about using redirection symbols with commands, see Chapter 2, “About Commands.”

Examples:

The following command reads the file *expenses.txt*, sorts it in reverse order, and displays it on your screen:

```
sort /r < expenses.txt
```

The following command pipes the output of the **dir** command to the **sort** filter. This filter sorts the directory listing starting with column 14 (the column in the directory listing that contains the file size) and sends the output to the screen. The result is a directory, sorted by file size:

```
dir | sort /+14
```

The following command does the same thing as the previous one, except that the **more** filter gives you a chance to read the sorted directory one screen at a time:

```
dir | sort /+14 | more
```

Sorting a file

Subst

Subst**Purpose:**

Substitutes a string for a path.

Syntax:

subst [*drive: drive:path*]

or

subst *drive: /d*

Comments:

The **subst** command lets you associate a path with a drive letter. This drive letter then represents a *virtual drive* because you can use the drive letter in commands as if it represented an actual physical drive.

When MS-DOS finds a command that uses a virtual drive, it replaces the drive letter with the path and treats that new drive letter as though it belonged to a physical drive.

If you type the **subst** command without options, MS-DOS displays the names of the virtual drives in effect.

Use the */d* switch to delete a virtual drive.

Notes:

The following commands do not work on drives used in the **subst** command (or the **join** command):

chkdsk	label
diskcopy	recover
fdisk	sys
format	

Examples:

The following command creates a virtual drive, drive Z, for the pathname *b:\user\betty\forms*:

```
subst z: b:\user\betty\forms
```

Note

This example assumes that you have included the line, **lastdrive=z**, in your *config.sys* file.

Now, instead of typing the full pathname, you can get to this directory by simply typing the name of the virtual drive:

Making a pathname alias

Creating a virtual drive

Sys

Sys (System)

Purpose:

Transfers MS-DOS system files from the disk in the default drive to the disk in the specified drive.

Syntax:

sys drive:

Comments:

The **sys** command updates your system files on a disk. You must type a drive letter following the **sys** command.

Updating your system

Notes:

- The transferred files are copied in the following order:

`io.sys`
`msdos.sys`

Io.sys and *msdos.sys* are both hidden files that *do not appear* when you type the **dir** command.

- **Sys** does not transfer the *command.com* file (the command processor). So to transfer *command.com* to the target disk, you must use the **copy** command.
- MS-DOS system files are no longer required to be contiguous. This means that you do not have to reformat your disk when you want to copy a new version of MS-DOS on a disk containing system files for MS-DOS version 3.3 or earlier.
- **Sys** doesn't work on drives involved with **subst** or **join** commands.
- **Sys** does not work on a network.

Examples:

If you want to copy the MS-DOS system files from your working directory to a disk in drive A, you would type

```
sys a:
```

Time

Time**Purpose:**

Allows you to enter or change the time known to the system.

Syntax:

time [*hours:minutes[:seconds [.hundredths]]*]

Comments:

MS-DOS keeps track of time in a 24-hour format, and uses the time information to update the directory whenever you create or change a file.

The **time** command without options displays the current time, and gives you an opportunity to change it:

```
Current time is hh:mm:ss.cc
Enter new time:_
```

If you do not want to change the time shown, simply press the **ENTER** key.

If you do want to change the time, type in a new value in the 24-hour clock format. The following are valid values:

```
hours = 0-23
minutes = 0-59
seconds = 0-59
hundredths = 0-99
```

Separate these elements (seconds and hundredths of seconds are optional) with the separator defined in the country-dependent information file. For the United States, use a colon (:).

You can also type the new time directly on the command line.

If you do not type a valid time, MS-DOS displays the following message and then waits for you to type a valid time:

```
Invalid time
Enter new time:_
```

Displaying the current time

Notes:

- You can change the **time** command format by changing the **country** command in the *config.sys* file. For more information, see Appendix B, "How to Configure Your System".
- The **time** command sets your computer's internal clock.
- You cannot specify a drive for this command.

Examples:

To reset the time of day on your computer's clock, you can type the **time** command by itself and MS-DOS will prompt you for the correct time. Or you can include the correct time when you type the command. For example, if you want to set your computer's clock at 1:36 p.m., you could type the following command:

```
time 13:36
```

Tree

Tree**Purpose:**

Graphically displays the path (and, optionally, lists the contents) of each directory and subdirectory on the given drive.

Syntax:

`tree [pathname:] [/f] [/a]`

Comments:

Finding directory names

The `tree` command lists the full path of each directory, along with the names of their subdirectories. If you specify only a drive name, the full tree structure is displayed starting from the root directory of that drive. If you specify a specific directory, the tree structure starting from that directory is displayed.

The `tree` command accepts the following switches:

The Tree switches

Switch	Purpose
<code>/f</code>	Displays the names of the files in each directory.
<code>/a</code>	Instructs <code>tree</code> to use the graphic characters available on all code pages. Allows for faster printing.

Examples:

If you want to see names of all directories and subdirectories on your computer, simply type

```
tree
```

If you also want to see, one screen at a time, the files in all the directories on drive C, you could type

```
tree c: /f | more
```

Printing a Tree listing

To print that same list on a printer, use the following command:

```
tree c: /f > prn
```

Type

Type

Purpose:

Displays the contents of a text file on the screen.

Syntax:

type [*drive:*]*filename*

Comments:

You can use the **type** command to view a text file without modifying it. (Use **dir** to find the name of a file and **Edlin** to change the contents of a file.)

Notice that when you use **type** to display a file that contains tabs, all the tabs are expanded to the current setting for tabs (generally eight spaces wide). Also, if you try to display a binary file or a file created by an application program, you may see strange characters on the screen, including bells, formfeeds, and escape sequence symbols.

Examples:

If you want to display the contents of a file called *holiday.mar*, you would type the following command:

```
type holiday.mar
```

If the contents of the file you wish to display are fairly long, you could use a command like this to display the file's contents one screen at a time:

```
type holiday.88 | more
```

Displaying a file

Ver

Ver**Purpose:**

Prints the MS-DOS version number.

Syntax:

ver

Comment:

If you want to know what version of MS-DOS you are using, you simply type the **ver** command. The version number will then be displayed on your screen.

Example:

When you type the **ver** command, the following message is displayed:

MS-DOS Version 4.0

**Displaying the MS-DOS
version**

Verify

Verify

Purpose:

Turns the verify switch on or off when writing to a disk.

Syntax:

verify [on]

or

verify [off]

Comments:

You can use this command to verify that your files are written correctly to the disk (no bad sectors, for example). MS-DOS verifies the data as it is written to a disk. You will receive an error message only if MS-DOS is unable to successfully write your data to a disk.

Notes:

This command has the same purpose as the /v switch in the **copy** command.

Examples:

If you want to know the current setting of **verify**, use the **verify** command without an option:

```
verify
```

Verify on remains in effect until a program changes it, or until you type the following:

```
verify off
```

Vol

Volume**Purpose:**

Displays the disk volume label and serial number, if they exist.

Syntax:

vol[*drive:*]

Comments:

This command displays the volume label of the disk in a specific drive. If you do not type a drive letter, MS-DOS displays the volume label of the disk in the default drive.

Notes:

- You cannot specify a drive name before this command. For example, if you typed *b:vol c:*, MS-DOS would display an error message because of the reference to drive B. MS-DOS assumes that this command resides on the drive you are working from.
- For more information about how MS-DOS uses volume labels, see the **label** and **format** commands in this chapter.

Examples:

Displaying a volume label

If you want to find out what the volume label is for the disk in drive A, you would type the following:

```
vol a:
```

If the volume label is "DOS 4-0" and the volume serial number is 2224-18C8, MS-DOS responds by displaying the message

```
Volume in drive A is DOS 4-0  
Volume serial number is 2224-18C8
```

If the disk in drive A does not have a volume label or volume serial number, MS-DOS responds by displaying the message:

```
Volume in drive A has no label.
```

Xcopy

Xcopy

Purpose:

Copies files and directories, including lower level directories, if they exist.

Syntax:

```
xcopy [drive:]pathname [drive:][pathname] [/a]/[d:date]  
[e]/[m]/[p]/[s]/[v]/[w]
```

or

```
xcopy drive:[pathname] [drive:][pathname] [/a]/[d:date]  
[e]/[m]/[p]/[s]/[v]/[w]
```

Comments:

The first set of *drive:* and *pathname* parameters specifies the source file or directory that you want to copy. The second set names the target. You must include at least one of the source parameters. If you omit the target parameters, **xcopy** assumes you want to copy the files to the default directory.

If you do not specify the *pathname* option, **xcopy** uses the default directory with the default filename, *.*.

The **xcopy** command accepts the following switches:

Switch	Purpose
/a	Copies source files that have their archive bit set. Does not modify the archive bit of the source file. For information on how to set the archive attribute, see the attrib command.
/d:date	Copies source files modified on or after the specified date. Notice that the date format may vary depending on the country code that you are using. For more information, see the date command.
/e	Copies any subdirectories, even if they are empty. You must use this switch with the /s switch.
/m	Same as the /a switch, but after copying a file, it turns off the archive bit in the source file. For information on how to set the archive attribute, see the attrib command.

The Xcopy switches

Switch	Purpose
/p	Prompts you with "(Y/N?)", allowing you to confirm whether you want to create each target file.
/s	Copies directories and lower level subdirectories, unless they are empty. If you omit this switch, xcopy works within a single directory.
/v	Causes xcopy to verify each file as it is written to the target to make sure that the target files are identical to the source files.
/w	Causes xcopy to wait before it starts copying files. Xcopy displays the following message: Press any key when ready to start copying files You must press a key to continue, or press CONTROL+C to abort the xcopy command.

Notes:**Copying to a disk with a different format**

- If you have a disk that contains files in subdirectories and you want to copy it to a target disk that has a different format, you should use the **xcopy** command rather than **diskcopy**. Since the **diskcopy** command copies disks track- by-track, it requires your source and target disks to have the same format.
- If **xcopy** encounters an error, it returns one of the following exit codes:

Xcopy exit codes

Code	Function
0	Copy without error
1	No files found to copy
2	CONTROL+C entered by user to terminate xcopy
4	Initialization error There is not enough memory-invalid drive or command line syntax, file not found, or path not found.
5	Int 24 error occurred The user aborted from INT 24 error reading or writing disk.

You can test for these codes by using the *errorlevel* condition of the batch processing **if** command.

Examples:

The following example copies all the files and subdirectories (including any empty subdirectories) on the disk in drive A to the disk in drive B:

```
xcopy a: b: /s /e
```

During the copy operation, the xcopy command may prompt you to specify whether the target is a file or a directory.

Using the Xcopy command

10/10/10

5

6

4 Batch Processing

In this chapter, you will learn

- How to create a batch file
- How an *autoexec.bat* file works
- How to use replaceable parameters in a batch file
- How to run a batch file

Note

If you are not writing batch programs, you do not need to read this chapter.

Why Use Batch Files?

You may often find yourself repeatedly typing the same sequence of commands to perform some common task. With MS-DOS you can put this command sequence into a special file called a *batch file*, and then run the whole sequence of commands by simply typing the name of the batch file. Notice that you don't need to type the batch file's extension, even though all your batch files must include the *.bat* extension in their filenames.

MS-DOS performs these "batches" of your commands just as if you had typed them from the keyboard. This is called *batch processing*. By using a batch file, you only have to remember to type one command, instead of several. In effect, you use batch files to create personalized commands.

How to Create Batch Files

You can create a batch file by using **Edlin** (the MS-DOS line editor), by using the **copy** command, or by using a word processor that saves files as ASCII text. If you want to create files with **Edlin**, you should refer to Chapter 6 for more information. The examples in this chapter show you how to use the **copy** command to create batch files.

Creating a batch file

Suppose, for example, that you want to create a batch file to format and check a new disk. To do this, you simply follow these steps:

- 1 First, type the following:

```
copy con checknew.bat
```

Press **ENTER**. This command tells MS-DOS to copy the information from the console (keyboard) to the file *checknew.bat*

- 2 Next, type the following lines, pressing the **ENTER** key after each:

```
rem This is a file to format and  
rem check new disks.  
rem It is named CHECKNEW.BAT.  
pause Insert new disk in drive B:  
format b: /v  
chkdsk b:
```

- 3 After the last line, press **CONTROL+C** and then press **ENTER** to save the batch file. MS-DOS displays the message "1 File(s) copied" to show that it created the file.

Running a batch file

- 4 Now, to execute the file, simply type the following command:

```
checknew
```

The result is the same as if the lines in the *.bat* file were typed from the keyboard as individual commands.

About Batch Processing

Here are a few things you should know before you run a batch process with MS-DOS:

- You must name each batch file with an extension of *.bat*.
- To execute a batch file, you type only its filename and not the extension.
- If you press **CONTROL+C** while the batch file is running, MS-DOS asks you to confirm that you want to terminate the batch process.
- If you remove the disk that contains a batch file being run, MS-DOS prompts you to reinsert the disk so that it can continue processing the file.

- You can specify the name of another batch file as the last command in a batch file. This feature allows you to call one batch file from another when the first has finished.
- You can use any of the redirection symbols (< > >>) in a batch file. For more information on using these symbols, see Chapter 2, "About Commands".
- You may use an @ character at the front of a command line in a batch file to prevent that line from echoing.
- You can use the pipe symbol (|) in a batch file.
- Setting the directory or drive affects every subsequent command in the batch file
- Setting environment strings also affects every subsequent command in the batch file.

Note

If you have more than one external command with the same name, MS-DOS will run only one of them, according to the following order of precedence: *.com*, *.exe*, *.bat*.

Suppose, for example, that your disk includes the files *format.com* and *format.bat*. If you were to type the external command *format*, MS-DOS would always run the program *format.com* first. To run the batch file *format.bat*, you would have to place it in a separate directory and give a path along with the external command.

For example, if your *format.bat* file is in a directory named `\commands\batch` and you want to run it, you would type the following:

```
\commands\batch\format
```

The Autoexec.bat File

An *autoexec.bat* file lets you run programs automatically when you start MS-DOS. This can be useful when you want to run a specific application under MS-DOS, and when you want MS-DOS to execute a batch program each time you start your computer. By using an *autoexec.bat* file, you can avoid loading two separate disks just to perform these tasks.

When you start your computer, MS-DOS searches the root directory of the default disk drive for a file named *autoexec.bat*. If it finds the *autoexec.bat* file, MS-DOS immediately processes it, bypassing the date and time prompts. If MS-DOS does not find an *autoexec.bat* file, then the date and time prompts appear automatically.

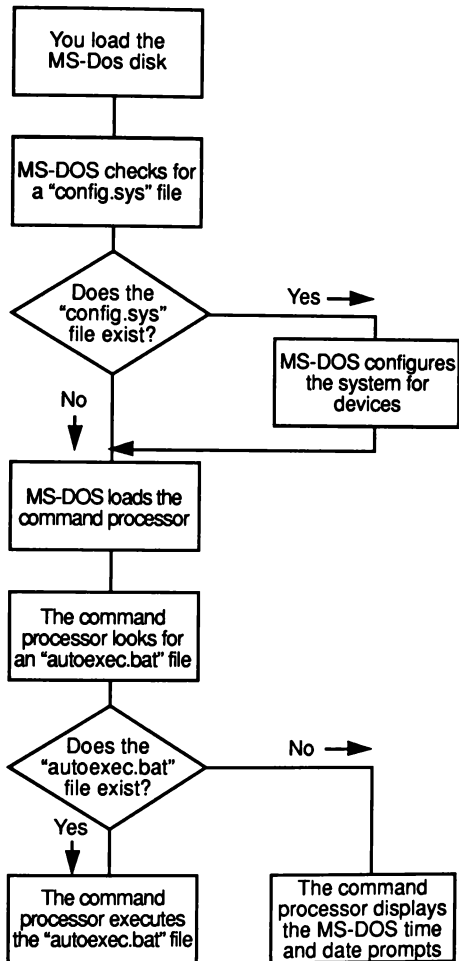
What Is an autoexec.bat file?

Hint

MS-DOS does not prompt you for a current date and time unless you include the **date** and **time** commands in your *autoexec.bat* file.

It's a good idea to add these two commands to your *autoexec.bat* file, since MS-DOS uses this information to keep your directory current. See Chapter 3, "MS-DOS Commands", for more information on the **date** and **time** commands.

The following figure shows what happens when you start MS-DOS:



How to Create an Autoexec.bat File

There are many things you can do with an *autoexec.bat* file to help you use MS-DOS more efficiently. For instance, you will probably want to set the time and date, your path, and any other options that you plan to use on a regular basis.

Note

When you create your *autoexec.bat* file, you must put it in the root directory of your MS-DOS disk.

If, for example, you want to automatically load GW-BASIC and run a program called *menu* each time you start MS-DOS, you could create an *autoexec.bat* file as follows:

- 1 Type the following command and then press ENTER:

```
copy con autoexec.bat
```

This command tells MS-DOS to copy what you type from the keyboard into the *autoexec.bat* file. Notice that you must put the *autoexec.bat* file in the root directory of your MS-DOS disk.

- 2 Now type the following lines:

```
date  
time  
path=c:\;c:\bin;a:\  
prompt [$p]  
cls  
gwbasic menu
```

- 3 After the last line, press CONTROL+Z and press ENTER to copy these lines into the *autoexec.bat* file
- 4 The *menu* program will now run automatically whenever you start MS-DOS.

Once your *autoexec.bat* file is set up as in this example, it will perform the following actions when you start MS-DOS: it will ask you to enter the date and time; it will set your command search path; and it will set your prompt to display the default drive and directory.

Finally, the *autoexec.bat* file will clear the screen and tell MS-DOS to load GW-BASIC and run the *menu* program. To run your own GW-BASIC program, type its name in place of *menu* in the example. In addition to GW-BASIC programs, you can also put any MS-DOS command or series of commands in the *autoexec.bat* file.

Creating an autoexec.bat file

How to Create a Batch File with Replaceable Parameters

There may be times when you want to create a program and run it with different sets of data. These data may be stored in various MS-DOS files.

With MS-DOS, you can create a batch (*.bat*) file with *replaceable* (dummy) *parameters*, where a *parameter* is a command option that you define. These parameters, named %0—%9, hold the places for the values that you supply when you give the batch command.

Replaceable parameters make batch files more flexible and easy to use. For example, you can create a batch file called *sorter.bat* that sorts a file containing a specific sequence of characters or strings. Each time you run the *sorter* batch file, you tell MS-DOS which string you want, which file to search to find that string, and which temporary file to use for sorting. *Sorter* would then print the resulting list on the printer.

- 1 To create the *sorter.bat* file, type the following command and then press the ENTER key:

```
copy con sorter.bat
```

- 2 Now type the following lines:

```
type %2 | find "%1" > %3
type %3 | sort > prn
del %3
```

- 3 To save the batch file, press CONTROL+Z and then ENTER .

The batch file *sorter.bat* now consists of three command lines and is on the disk in the default drive.

When you execute the file, MS-DOS sequentially replaces %1, %2, and %3 with the parameters you supply. If you use the dummy parameter %0, MS-DOS always replaces it with the drive name (if specified) and the filename of the batch file (for example, *sorter*).

Notes

- You can specify up to ten replaceable parameters (%0—%9). If you want to specify more than ten, refer to the shift command later in this chapter.
- If you use the percent sign as part of a filename *within a batch file*, you must type it twice. For example, to specify the file *abc%.exe*, you must type it as *abc%%.exe* in the batch file.

Using replaceable parameters

Using named parameters

How to Use Named Parameters in a Batch File

In addition to the ten, numeric replaceable parameters, you can use named parameters in batch files. Named parameters allow you to define your replaceable parameters by name instead of by number. When referenced in a file, the name of a replaceable parameter is placed between two percent signs.

Named parameters differ from replaceable parameters in that you do not specify their values on the command line. Instead, MS-DOS retrieves the value of each named parameter from its environment.

You can use the MS-DOS set command to set the value of a named parameter before you run your batch file, or you can include the set command in your batch file.

For example, suppose that you want to create a batch file called *mydel.bat*, that moves a file you want to delete into a separate directory. You might want to use this method to make sure that you don't delete files accidentally from an important directory.

The file *mydel.bat* might contain the following lines:

```
echo off
echo Before you use this batch file, you must
echo specify the directory by typing the
echo following command at the MS-DOS prompt::
echo set deldir=directory
echo Press Control-C to exit if you haven't
echo set deldir or if deldir does not exist..
pause
copy %1 %deldir%
del %1
dir /w %deldir%
echo All done.
```

For this *mydel.bat* file to work, you must create a directory named *deleted*. The following command line sets the directory name to *deleted*:

```
set deldir=deleted
```

Now, to move the file *report23.jun* to the *deleted* directory, type the following:

```
mydel report23.jun
```

The batch file automatically replaces the *%deldir%* parameter with the directory name *deleted*.

You may find named parameters easier to use than replaceable parameters because you won't have to include as much information on the command line. For example, with *mydel.bat*, you don't have to type the directory name on the command line, yet you can change the name of the directory without having to edit the batch file.

How to Run a Batch File

To run the batch file *sorter.bat*, type the batch filename followed by the parameters that you want MS-DOS to substitute for %1, %2, and %3.

Running a batch file

Suppose that on the disk in drive A you have a file that lists your customers' names and regions. The file might look something like this:

```
Shores, Betty north
Moynihan, Ann south
Kraig, Heidy north
Martin, Pete east
Lennon, Patrick south
Pai, Fernando north
Evans, Rick west
Moss, Melissa north
```

If you want to print an alphabetical list of the customers in the north, you can run the *sorter* batch file, with the appropriate parameters, by typing the following command and then pressing the ENTER key:

```
sorter north a:customer temp.fil
```

The output on the printer should look like this:

```
Kraig, Heidy north
Moss, Melissa north
Pai, Fernando north
Shores, Betty north
```

The following table shows how MS-DOS replaces each of the parameters in the previous example:

Batch filename	(%0)	sorter
Parameter1	(%1)	north
Parameter2	(%2)	a:customer
Parameter3	(%3)	temp.fil

The result is the same as if you had typed each of the commands in *sorter* with its parameters, as follows:

```
type a:customer | find "north" > temp.fil
type temp.fil | sort > prn
del temp.fil
```

Using the batch file, however, saves typing time and is much easier to remember.

How to Use Temporary Files

Using temporary files

When using batch files, you may often want to use a temporary file to hold your work. You could use the same name each time you wanted to use a temporary file.

However, if you are using more than one batch file that uses the same temporary file, you might lose the contents of this temporary file. To avoid this problem, you should use a replaceable parameter to specify the name of the temporary file. Then each time you run the batch file, you'll be able to substitute a unique filename and you won't have to worry about information from one batch file getting into another.

It's also a good idea to delete temporary files once you finish using them. Otherwise, these files would eventually take up all the space on your disk.

Batch Processing Commands

Now that you have seen some of the capabilities of batch files, in this section you'll find out how to add power and flexibility to your batch programs by using batch processing commands. The following table lists these batch commands and describes what they do:

Using the batch processing commands

Command	What it does
call	Calls one batch file from another without ending itself
echo	Turns the batch file echo feature on or off, or displays the current setting
for	Performs a command for a set of files
goto	Processes commands starting with the line after the specified label
if	Performs a command if a condition is met
pause	Pauses during the processing of a batch file
rem	Displays a comment in a batch file
shift	Increases the number of replaceable parameters in a batch process

Batch commands are also internal commands, so each one is marked with the internal command icon (the boxed letter I) used in Chapter 3, "MS-DOS Commands".

Call**Call****Purpose:**

Calls one batch file from another without ending itself.

Syntax:

call [*drive:*][*path*] *batchfile* [*argument*]

where:

batchfile is the batchfile you want to call.

argument is the command in this batch file that will be run following *batchfile*.

Comments:

- The **call** command is used within one batch file to call another one. *Batchfile* must have a filename extension of *.bat*.
- When *batchfile* terminates, the calling batch file resumes running at *argument*. If *argument* is omitted, the calling batch file resumes running at the command immediately following the **call** command.

Notes:

- Do not use pipes and redirection symbols with the **call** command.
- A batch file can make a recursive call to itself, but there should be a termination condition that is eventually met.

Examples:

To run the *checknew.bat* file from another batch file, you would use the following command within the first batch file:

```
call checknew
```

Echo

Purpose:

Turns the batch echo feature on and off.

Syntax:

echo [on]

or

echo [off]

or

echo [*message*]

Comments:

Normally, commands in a batch file are displayed ("echoed") on the screen when they are received by MS-DOS. You can turn off this feature by using the **off** option with the **echo** command. Similarly, you can turn the echo feature back on by using the **on** option with **echo**.

If you do not specify **on** or **off**, **echo** displays the current setting.

The command, **echo message** (where *message* is a line of text), is only useful if **echo** is off and if you are using a batch file. If, in your batch file, you type the **echo** command followed by a message, you can print messages on your screen. You can also put several echo message commands in your batch file to display a message that is several lines in length.

An "@" character placed in front of a command line in a batch file prevents that line from echoing.

Examples:

The following is an example of a batch file message of more than one line:

```
echo off
echo This batch file
echo formats and checks
echo new disks.
```

If you want to turn **echo** off, and do not want the command itself to be echoed, include the @ character before the command line:

```
@echo off
```

Echo

Creating messages for a batch file

For**For****Purpose:**

Performs a command for a set of files.

Syntax:

for %*nc* in *set* do *command* (for batch processing)

for %*c* in *set* do *command* (for interactive processing)

Comments:

To avoid confusion with the %0—%9 batch parameters, the variable *c* can be any character except 0,1,2,3,...,9.

set is (*item**)

This command sequentially sets the %*nc* variable to each member of *set*, and uses the variable to evaluate *command*. If a member of *set* is an expression involving a wildcard (* or ?), then the variable is set to each matching *item* from the disk. In this case, only one such item is in *set*, so the command ignores any item other than the first.

Examples:

The following example assigns the variable %*f* to any files ending with *.asm in the working directory:

```
for %%f in ( *.asm ) do masm %%f
```

It then executes a command of the following form:

masm *filename*

Filename could be any one of the following:

```
invoice.asm  
receipts.asm  
taxes.asm
```

The following example assigns the variable %*f* to the files named *report*, *memo*, and *address*; it then deletes each of these files:

```
for %%f in (report memo address) do del %%f
```

You must use two percent signs (%%) so that one will remain after the batch parameter (%0-%9) processing is complete. If you had only %f, instead of %%f, then the batch parameter processor would see the %, look at f, decide that %f was an error (a bad parameter reference), and throw out the %f so that the for command would never see it.

Notice that if you are using the for command outside of a batch file, you should use only one percent sign.

Goto

Goto**Purpose:**

Processes commands starting with the line after the specified label.

Syntax:

goto [:]*label*

Comments:

Goto lets you take commands from the batch file beginning with the line after the *label*, where a *label* is defined as the characters following **goto**. This *label* may include spaces, but not other separators, such as semicolons or equal signs. If your batch file does not contain the *label*, the batch process terminates.

Note

Any line in a batch file that starts with a colon (:) is ignored during batch processing.

Examples:

Errorlevel processing

The following example sends the program processor to the label named *end*—only if no errors occur when you format the disk in drive A:

```
:begin
echo off
format a: /s
if errorlevel 0 goto end
echo An error occurred during formatting.
:end
echo End of batch file.
```

If

If**Purpose:**

Performs a command based on the result of a condition.

Syntax:

if [**not**] *errorlevel number command*

or

if [**not**] *string1 = = string2 command*

or

if [**not**] *exist filename command*

Comments:

The **if** statement allows conditional execution of commands. When the condition is true, MS-DOS executes the *command*; otherwise, it ignores the *command*.

The conditions are described as follows:

Condition	Description
errorlevel number	True if, and only if, the previous program executed by <i>command.com</i> had an exit code equal to, or greater than, <i>number</i> . (When a program finishes, it returns an exit code via MS-DOS.) You can use this condition to perform other tasks that are based on the previous program's exit code.
string1 = = string2	True if, and only if, <i>string1</i> and <i>string2</i> are identical after parameter substitution. Strings may not contain separators, such as commas, semicolons, equal signs, or spaces.
exist filename	True if, and only if, <i>filename</i> exists.

If conditions

If you specify the **not** parameter, MS-DOS executes the *command* when the condition is false.

Notes:

For more information about exit codes returned by an MS-DOS command, see the specific command in Chapter 3, "MS-DOS Commands".

Examples:

Displaying a conditional message

The following example prints the message "can't find datafile" if the file *product.dat* does not exist on the disk:

```
if not exist product.dat echo can't find datafile
```

Errorlevel processing

The following example sends the program processor to the label named *end*—only if no errors occur when you format the disk in drive A.

```
:begin
echo off
format a: /s
if errorlevel 0 goto end
echo An error occurred during formatting.
:end
echo End of batch file.
```

Pause

Purpose:

Suspends execution of a batch file.

Syntax:

pause [*comment*]

Comments:

When a batch file is running, you may need to change disks or perform some other action. The **pause** command suspends execution of the batch file until you press any key, unless you press the **CONTROL+C** key sequence.

When the command processor encounters **pause**, it prints the following message:

```
Strike a key when ready . . .
```

If you press **CONTROL+C**, MS-DOS displays the following message:

```
Terminate batch job (Y/N)?
```

If you type *Y* (for Yes) in response to this prompt, the batch file ends and control returns to the operating system. Therefore, you can use **pause** to divide a batch file into pieces that allow you to end the batch command file at any intermediate point.

The *comment* parameter is useful when you want to display a special message. Unless **echo** is off, **pause** displays this comment before the "Strike a key message".

Note The **pause** and comment line of your batch file will not appear if **echo** is off.

Examples:

Suppose you want a program to display a message that asks the user to change disks in one of the drives. To do this you might use the following command:

```
Pause Please put a new disk into drive A
```

If **echo** is on, this line will precede the "Strike a key" message when you run the batch file.

Pause

Using Pause messages in a batch file

Rem

Rem**Purpose:**

During execution of a batch file, **rem** displays remarks that are on the same line as the **rem** command in that batch file.

Syntax:

rem [*comment*]

Comments:

The *comment* parameter is a line of text that helps you identify and remember what your batch file does.

The only separators allowed in the *comment* are spaces, tabs, and commas.

In your batch file, you can use **rem** without a comment to add spacing for readability.

Note

If **echo** is off, the **rem** comment is not displayed.

Examples:

The following example shows a batch file that uses remarks for both explanation and spacing:

```
rem This file formats and checks new disks
rem It is named checknew.bat
rem
pause Insert new disk in drive B
format B: /v
chkdsk B:
```

Using remarks in a batch file

Shift

Purpose:

Lets you change the position of replaceable parameters in batch file processing.

Syntax:

shift

Comments:

You can use the **shift** command to change the positions of (replaceable) command line parameters.

Usually, command files are limited to handling ten parameters, %0 through %9. But by using the **shift** command, you can access more than ten parameters. This means that if there are more than ten parameters given on a command line, those that appear after the tenth (%9) will be shifted one at a time into %9.

You can use the **shift** command even if you have less than ten parameters.

Warning

There is no backward **shift** command. Once you have executed **shift**, you cannot recover the first parameter (%0) that existed before the **shift**.

Examples:

The following file, called *mycopy.bat*, shows how to use the **shift** command with any number of parameters. It copies a list of files to a specific directory.

```
rem mycopy.bat copies
rem any number of files
rem to a directory.
rem The command is
rem mycopy dir files
set todir = %1
:one
shift
if "%1"==" " goto two
copy %1 %todir%
goto one
:two
set todir=
echo All done
```

Shift

Shifting replaceable parameters

1000
1000
1000

5 MS-DOS Editing and Function Keys

In this chapter, you will learn about

- The MS-DOS editing and function keys
- The editing template
- The MS-DOS control characters

Special MS-DOS Editing Keys

Many operating systems handle command input differently than MS-DOS does. One difference in particular that sets MS-DOS apart is its set of special editing keys. For instance, with MS-DOS you don't have to type the same sequences of keys repeatedly, because the most recently typed command line is automatically placed in a special storage area called a *template*.

By using the template and the special editing keys, you can take advantage of the following MS-DOS features:

- You can repeat a command instantly by pressing two keys.
- If you make a mistake in a command line, you can edit and retry it without having to retype the entire line.
- With a minimum of typing, you can edit and execute a command line that is similar to a previous one.

How MS-DOS Uses the Template

When you type a command and press the ENTER key, MS-DOS automatically sends it to the command processor (*command.com*) for execution. At the same time, MS-DOS also saves a copy of this command in the template. You can then recall or modify the command by using the MS-DOS special editing keys.

The MS-DOS Editing Keys

Special editing functions

The MS-DOS editing keys provide you with a set of editing tools that can save you time. You can use them to correct typing mistakes, repeat frequently used commands, or create similar command lines. These keys are described briefly in the following table and more fully later in Chapter 6, "The Line Editor (Edlin)."

Key	Editing function
F1	Copies one character from the template to the command line.
F2	Copies characters up to the character specified in the template and puts these characters on the command line.
F3	Copies all remaining characters in the template to the command line.

Key	Editing function
DEL	Skips over (does not copy) a character in the template.
F4	Skips over (does not copy) the characters in the template up to the character specified.
ESC	Voids the current input and leaves the template unchanged.
INS	Enters/exits insert mode.
F5	Makes the new line the new template.
F6	Puts a CONTROL+Z (1AH) end-of-file character in the new template.

You'll use these special editing keys in conjunction with the template, which you'll learn to use in the next section.

How to Use the MS-DOS Template

Examples:

Creating a template

Suppose you want to see the directory information for a file named *invest.mnt*. To get this information you could type the following command:

```
dir invest.mnt
```

This command line (*dir invest.mnt*) is also saved in the template. If you want to repeat the command, just press two keys: F3 and ENTER.

MS-DOS displays the repeated command on the screen when you press F3, as shown below:

```
dir invest.mnt
```

Notice that when you press the F3 key, MS-DOS copies the contents of the template to the command line; pressing the ENTER key then sends the command line to the command processor for execution.

Changing the template

If you want to display information about a file named *invest.rpt*, you can use the contents of the template. Pressing F2, followed by the letter *m*, copies all characters from the template to the command line, up to but not including the *m*. MS-DOS displays

```
dir invest._
```

Notice that the underline is your cursor. Now type the letters *rpt* to get the following result:

```
dir invest.rpt_
```

The command line (*dir invest.rpt*) is now in the template and ready to be sent to the command processor for execution. To run the command, press the ENTER key.

Now, assume that you want to run the following command:

```
type invest.rpt
```

To do this, type the word *type* and then press the following sequence of keys: INS, SPACEBAR, F3, ENTER

Notice as you type that the characters appear directly on the command line, overwriting their corresponding characters in the template. Before you press the **INS** key, the word *type* replaces the word *dir* (and the space following it) in the template. After you press the **INS** key, this automatic replacement feature is turned off.

To insert a space between the word *type* and the filename *invest.rpt*, you pressed **INS** and then the **SPACEBAR**. Finally, to copy the rest of the template to the command line, you pressed **F2** and then the **ENTER** key.

The command line *type invest.rpt* has been processed by MS-DOS, and the template now looks like this:

```
type invest.rpt.
```

If you had misspelled *type* as *pyte*, for example, a command error would have occurred. Still, instead of throwing away the whole command, you could save the misspelled line before pressing the **ENTER** key. You can do this by pressing the **F5** key *before* the **ENTER** key, creating a new template:

```
pyte invest.rpt
```

Then you can use the **DEL** and **INS** keys to get the same result, as follows:

```
DEL DEL F1 INS yp F3
```

Correcting errors in the template

To illustrate how the keys you type affect the command line, compare the key pressed with its result (shown beneath it along with a description of the effect).

DEL	
<code>_</code>	Skips over 1st template character
DEL	
<code>_</code>	Skips over 2nd template character
F1	
<code>t_</code>	Copies 3rd template character
INS yp	
<code>t yp_</code>	Inserts two characters, y and p
F3	
<code>type invest.rpt_</code>	Copies rest of template

Notice that **DEL** does not affect the command line. Instead, it affects the template by deleting the first character. Similarly, pressing the **F1** key deletes characters in the template, up to, but not including, a given character.

These special editing keys do give you more power and flexibility when you are typing. But in addition to these keys, MS-DOS also has control characters that help you control the output from a command, or control the contents of the current command line. The next section describes how to use the MS-DOS control characters.

The MS-DOS Control Characters

Using the MS-DOS control characters

A control character affects the command line in a special way. For example, you use **CONTROL+C** to stop running the current command, and you use **CONTROL+S** to suspend the screen output from a command.

Note

When you type a control sequence, such as **CONTROL+C** you must hold down the **CONTROL** key and then press the **C** key.

The following table shows the MS-DOS control characters and describes what they do:

Control character	What it does
CONTROL+C	Aborts the current command.
CONTROL+H	Removes the last character from a command line, and erases that character from the terminal screen.
CONTROL+J	Inserts a physical end-of-line, but does not empty the command line. Use the LINEFEED key to extend the current logical line beyond the physical limits of the terminal screen.
CONTROL+N	Causes echoing of output to a lineprinter.
CONTROL+P	Causes terminal output to a lineprinter.
CONTROL+S	Suspends output display on the screen. Press CONTROL+S again to resume.
CONTROL+X	Cancels the current line, empties the command line, and then outputs a backslash (\), ENTER , and LINEFEED . CONTROL+X does not affect the template used by the special editing commands.

Now that you have been introduced to the special editing and function keys of MS-DOS, in the next chapter you'll learn how to use them with Edlin, the MS-DOS line editor.

What comes next?

6 The Line Editor (Edlin)

Edlin is the MS-DOS line editor that you can use to create text files and save them on your disks. **Edlin** also helps you to update existing files by deleting, changing, and inserting lines in files. And even though it isn't a word processor, **Edlin** does make it easy for you to create and revise files such as memos, letters, reports, or GW-BASIC programs.

What can Edlin do?

- In this chapter, you will learn
- How to start **Edlin**
- How to end **Edlin** and save edits
- How to use the MS-DOS special editing keys with **Edlin**
- How to use **Edlin** commands

How Edlin Works

Edlin divides the text from a file into lines, each line containing up to 253 characters. It gives each line a number and always numbers the lines consecutively. But even though you see these line numbers on the screen when you use **Edlin**, they are not part of the file.

When you insert lines of text in a file, the line numbers after the inserted text are automatically adjusted. Similarly, when you delete lines in a file, the line numbers following the deleted text are automatically renumbered.

How to Start Edlin

Starting Edlin

To start **Edlin**, type a command of the following form:

edlin *filename*

where *filename* is the file you want to edit. If you are creating a new file, *filename* should be the name or pathname of the file you want to create. If **Edlin** does not find this file on the default disk drive, it creates a new file with the name or pathname that you specify. For example, if you want to create a file called *budget.jun*, you would type the following command and then press the **ENTER** key:

```
edlin budget.jun
```

Creating a new file with Edlin

Once you type the command, **Edlin** displays the following:

```
New file
*
_
```

Notice that the **Edlin** prompt is an asterisk (*).

To begin entering text, you must type an **i** (insert) command to insert lines. The **i** command is discussed later in this chapter. For now you can type lines of text into your file, or use any of the **Edlin** commands, which are also discussed later in this chapter.

Note

Be sure to press the **ENTER** key at the end of each line.

Suppose you want to edit an existing file called *budget.may*. To do this you would type the following:

```
edlin budget.may
```

When **Edlin** finds the *budget.may* file, it loads it into memory. If your computer has enough memory to load the entire file, **Edlin** displays the following message:

```
End of input file
*
```

You can then edit the file by using **Edlin** commands.

If the file is too large to be loaded into memory, **Edlin** loads lines from the file until memory is 3/4 full, and displays the asterisk (*) prompt. You can then edit the portion of the file that is in memory.

To edit the rest of the file, you must save some of the edited lines on a disk to free memory. **Edlin** will then be able to load the remaining unedited lines from a disk into memory. For more information on editing large files, see the **w** (write) and **a** (append) commands later in this chapter.

How to End Edlin and Save Your Changes

When you finish your editing session and the cursor is at the asterisk (*) prompt, you can save your original file and the updated (new) file by using the **e** (end) command, discussed later in this chapter. **Edlin** renames your original file with the extension *.bak*, and saves the updated file with the filename and extension you gave when you started **Edlin**.

Saving a file

Warnings

- You cannot update a file with an extension of *.bak* because when you try to save your file, **Edlin** always saves the original file as *.bak*, overwriting any existing *.bak* file. Thus, any changes you made to the existing *.bak* file using **Edlin** are lost. If you need to update a *.bak* file, rename it with another extension (using the MS-DOS **ren** command discussed in Chapter 3, "MS-DOS Commands"), and start **Edlin** by using the new filename.

- If a *.bak* file already exists and has read-only permission, the updated file will be saved by using the *e* (end) command, but the original file will not be saved in a *.bak* file. Edlin detects this condition when you first start to edit a file and displays the following message:

WARNING! Backup is Read-only - backup will not be made

Special Editing Keys

Using the editing keys with Edlin

To edit your text files, you can also use the special editing keys and template introduced in Chapter 5, "MS-DOS Editing and Function Keys".

The following table summarizes the commands, codes, and functions of the special editing keys. Descriptions of these special editing keys follow the table.

Key	Purpose
F1	Copies one character from the template to the new line.
F2	Copies all characters from the template to the new line, up to the character specified.
F3	Copies all remaining characters in the template to the screen.
DEL	Does not copy (skips over) a character.
F4	Does not copy (skips over) the characters in the template, up to the character specified.
ESC	Clears the current input and leaves the template unchanged.
INS	Enters/exits the insert mode.
F5	Makes the new line the new template.
BKSP	Deletes a character from the command line and places the cursor back one character in the template.

The following pages describe how to use each of the MS-DOS editing keys with Edlin.

The F1 Key

When you press the **F1** key, **Edlin** copies one character from the template to the current line, and turns the insert mode off. As an example of how to use the **F1** key with **Edlin**, type the following lines:

```
1:*Sharpe Office Supplies.
2:*_
```

At the beginning of the editing session, the cursor (shown by the underline) is at the beginning of the line. When you press the **F1** key, **Edlin** copies the first character, **S**, to line 2 as shown here:

```
2:*s_
```

Each time you press the **F1** key, one more character appears:

```
F1
2:*Sh_
```

```
F1
2:*Sha_
```

```
F1
2:*Shar_
```

Copying one character

The F2 Key

When you press the **F2** key, **Edlin** copies all the characters, up to a given character, from the template to the current line. The given character is the one that you type immediately after pressing **F2**. **Edlin** does not copy or display the given character on the screen, but it does copy and display the characters from the template up to the position of that character. Of course, if the template does not contain the character, **Edlin** does not copy anything.

If you use the **F2** key, you automatically turn off the insert mode.

As an example of how to use the **F2** key with **Edlin**, if you type *Sharpe Office Supplies.* on line 1 and then press **F2**, followed by the letter *c*, **Edlin** copies the characters up to the *c* in the word *Office*.

```
F2 c
2:*Sharpe Offi_
```

Copying multiple characters

Copying the template

The F3 Key

When you press the **F3** key, Edlin copies the remaining characters in the template to the current line. No matter where the cursor is when you press the **F3** key, Edlin displays the rest of the line and leaves the cursor at the end of the line. As an example of how to use the **F3** key with Edlin, if you type *Sharpe Office Supplies.* on line 1 and press **F3**, Edlin copies the characters in the template (from line 1) to the line with the cursor (shown in line 2):

```
F3  
2:*Sharpe Office Supplies._
```

Also, this command automatically turns off the insert mode.

Skipping one character

The DEL (Delete) Key

Each time you press the **DEL** key, Edlin skips over and does not copy the next character in the template. The action of the **DEL** key is similar to that of the **F1** key, except that **DEL** skips a character in the template instead of copying it to the current line. Thus, if the template already contains "Sharpe Office Supplies". and you press the **DEL** key, Edlin skips over the first character, *S*. The cursor does not move as Edlin changes the template. So to see how much of the line has been skipped over, press the **F3** key. This action moves the cursor past the last character of the line:

```
DEL  
F3  
2:*harpe Office Supplies._
```

Skipping multiple characters

The F4 Key

When you press the **F4** key, Edlin skips over all characters up to a given character in the template. Edlin does not copy or display any of the characters up to and including the given character. Of course, if the template does not contain that character, Edlin does not skip any characters.

Notice that the action of the **F4** key is similar to that of the **F2** key, except **F4** skips over characters in the template instead of copying them to the current line. Using the same template example, "Sharpe Office Supplies", the **F4** key followed by the letter *c* causes Edlin to skip over all the characters in the template up to the *c* in the word *Office*. The cursor does not move as Edlin changes the template. To see how much of the line has been skipped over, press the **F3** key to copy the template. This action displays the rest of the line and moves the cursor to the end of the line:

```
F4 C
F3
2:*ce Supplies._
```

The ESC (Escape) Key

When you press the **ESC** key, Edlin clears the current line and leaves the template unchanged. **ESC** also prints a backslash (****), **ENTER**, and **LINEFEED**, and turns the insert mode off. The cursor is at the beginning of the line. If you then press the **F3** key, Edlin copies the template to the current line, making it appear as it was before you pressed the **ESC** key. For example, type the following lines:

```
1: Sharpe Office Supplies.
2:*The World Leader_
```

To cancel the current line, line 2, press **ESC**. Notice that a backslash appears on line 2 to tell you it has been canceled:

```
ESC
2:*The World Leader\
```

Press the **ENTER** key to keep line 1, or to perform any other editing functions. Now if you press **F3**, Edlin copies the original template to the line:

```
F3
2:Sharpe Office Supplies.
```

Clearing current input

Entering Insert mode

The INS Key

The **ins** key toggles between insert mode and replace mode. When you start **Edlin**, you are automatically in replace mode. The first time you press the **ins** key, **Edlin** enters insert mode. In insert mode, the cursor in the template does not move, but in the current line it moves as you insert each character. When you finish inserting characters and press the **ins** key again, **Edlin** re-enters replace mode with the cursor at the same character in the template as when you entered insert mode.

Creating a new template

The F5 Key

When you press the **f5** key, **Edlin** copies the current line to the template and deletes the previous contents of the template. Pressing **f5** also displays an @ ("at" symbol), and outputs an **ENTER** and a **LINEFEED**. When you press **f5**, **Edlin** empties the current line and turns off insert mode.

Note

f5 performs the same function as the **esc** key, but it changes the template, printing an @ instead of a backslash.

As an example of how to use the **F5** key with **Edlin**, type the following line:

```
1:*Sharpe Office Supplies.
2:*
```

Remember that at the beginning of the editing session, the cursor (shown by the underline) is at the beginning of the line. Now type the following sequence of keys and words (the results are shown below each key sequence):

```
f2 c
2:*Sharpe Offi_
```

```
INS cial
2:*Sharpe Official_
```

```
INS Sharpeware.
2:*Sharpe Official Sharpeware._
```

At this point, suppose you want to add a word at the beginning of this line, but you don't want to backspace and retype the whole line. Just press the **F5** key to put the current line into the template:

```
1: Sharpe Office Supplies.
2:*Sharpe Official Sharpeware.@
```

The **@** shows that this new line is now the new template. To add the word *Introducing*., followed by a space, at the beginning of the line, press **INS** and type the following sequence of keys and words (the results are shown below each key sequence):

```
INS Introducing:
2:*Introducing: _
```

Then press the **F3** key to insert the contents of the template.

```
F3
2:*Introducing: Sharpe Official Sharpeware._
```

The BKSP (Backspace) Key

This key deletes a character from the command line and places the cursor back one character in the template.

Note

You can also press the **LEFT** direction key or **CONTROL+H** to backspace. For example, suppose that you type the following:

```
1:*Sharpe Office Supplier_
```

Then you realize that you meant to type *Supplies* instead of *Supplier* to correct this error, you would press the **BKSP** key followed by the letter *s*.

```
BKSP s
1:*Sharpe Office Supplies_
```

**Deleting one command
line character**

Using Edlin Commands

Edlin commands

Edlin includes several commands that help you to edit lines of your file. The following table summarizes these commands:

Letter	Purpose
a	Appends lines from disk to memory.
c	Copies lines.
d	Deletes line.
line	Edits a line or lines.
e	Ends editing session and saves edits.
i	Inserts lines of text.
L	Lists a range of lines.
m	Moves a range of text to a specified line.
p	Pages through a file 23 lines at a time.
q	Quits the editing session without saving the file.
r	Replaces text.
s	Searches for text.
t	Transfers the contents of another file into the file being edited.
w	Writes specified lines from memory to disk.

Some Tips for Using Edlin Commands

Once you have started editing a file with Edlin, you can use the Edlin commands to edit lines of text in the file. Here are a few things to remember when using Edlin commands:

- You can use pathnames in commands. For example, by typing the following command, you can edit a file named *report.may* in a subdirectory named *\sharpe\budget*:

```
edlin \sharpe\budget\report.may
```


- You can refer to lines with numbers relative to the current line (identified with an asterisk, *) To indicate lines before the current line, use a *minus* sign with a number; to indicate lines after the current line, use a *plus* sign with a number. For example, to list 10 lines before the current line, the current line, and 10 lines after the current line, you could type this command:

```
-10,+10L
```

Note

A capital L is used here for the L (list) command to avoid confusion with the number one. A small letter l would work just as well.

- Edlin ignores spaces between the line number and command. For the examples in this chapter, spaces are omitted.
- Generally, Edlin allows you to type one command after another on the same command line. However, if you want to use the **Edlin line** (edit) command to edit a specific line, you must separate the line number from the other commands with a semicolon. For example, the following command edits line 15, then displays lines 10 through 20 on the screen:

```
15;-5,+5L
```

- When using control key sequences, simply press and hold the **CONTROL** key followed by the control character (such as z or c or v). For example, if you want to search for a phrase that includes the control character **CONTROL+z**, you could use a command like this one:

```
smonthly budgetCONTROL+z-5,+5L
```

Notice that you do not type the letters **CONTROL** but instead press the **CONTROL** key as you press the z key.

- You can insert a control character, such as **CONTROL+c**, into text by using the quotation mark character, **CONTROL+v**, before it while in insert mode. **CONTROL+v** tells MS-DOS to recognize the next *capital* letter typed as a control character. For example, the following command finds the first occurrence of **CONTROL+z** (the end-of-file mark) in a file:

```
sCONTROL+ZZ
```

To insert **CONTROL+v** into the text, press **CONTROL+v** and type V.

Using control characters

- The **CONTROL+Z** character is usually an end-of-file identifier for **Edlin**. If you have **CONTROL+Z** characters elsewhere in your file, you must tell **Edlin** that these other control characters do not mean end-of-file. To tell **Edlin** to ignore the **CONTROL+Z** characters in the file and to show you the entire file, use the **/b** switch when you start **Edlin**. For example, the following command lets you start editing the file *macro.asm* and ignores any **CONTROL+Z** characters:

```
edlin macro.asm /b
```

Edlin Command Options

Using command options with Edlin

Many **Edlin** commands accept one or more options. The effect of a command option varies, depending on which command you use it with. The following list describes each option.

The Line Option

The *line* option is a line number that you type. Use a comma or space to separate the numbers from other line numbers, other options, and from the command.

You can specify *line* in one of three ways:

Type	How it works
<i>linenumber</i>	You may use any number less than 65,534. If you specify a number larger than the largest existing line number, then <i>line</i> refers to the line after the last line number.
period (.)	If you specify a period for <i>line</i> , it refers to the current line number. <i>The current line is the last line you edited, not necessarily the line you last displayed.</i> Edlin marks the current line with an asterisk (*) between the line number and the first character.
pound sign (#)	The pound sign indicates the line after the last line number in the file. If you type # for <i>line</i> , it is the same as typing the last line number plus one.

Type	How it works
ENTER key	If you type a command and then press the ENTER key without any of the line markers in this list, Edlin uses a default value for each command (default values may be different for each command).

The Question Mark Option

The question mark (?) option tells Edlin to ask you if the correct string has been found. You use the question mark only with the **r** (replace) and **s** (search) commands. Before continuing, Edlin waits for you to type a **Y** or press the ENTER key for a "Yes" response, or to press any other key for a "No" response.

The Text Option

The *text* option specifies text to be found, to be replaced, or to replace other text. Use the *text* option only with the **s** (search) and **r** (replace) commands.

The Edlin Commands

The remaining pages in this chapter describe the Edlin commands. Each description includes the correct usage (syntax) of a command. Also, each command description includes several comments and examples that offer advice, help, and even some shortcuts for using Edlin.

Appending a line

Edlin's Append Command: a**Syntax:**

[*n*]a

Comments:

If you are editing a large file that is too large to read into memory all at once, you can use the **a** (append) command. This command lets you read in portions of your file to memory as you need to work on them. The *n* parameter is the number of lines that you want to read into memory.

When you start **Edlin**, it reads as many lines as possible into memory. If the size of your file exceeds available memory, you must edit your file in stages. That is, after you have edited the first part of a large file, you must write lines that you have already edited onto your disk. Then you can load unedited lines from your disk into memory by using the **a** (append) command.

Notes:

If you do not specify the number of lines to append, **Edlin** adds lines to the available memory until it is 3/4 full, but does nothing if available memory is already 3/4 full. If available memory is already full, you may be able to free memory by quitting other applications that may be running or by restarting MS-DOS. Restarting MS-DOS clears memory being used by programs that remain resident in memory, even after they finish running.

After the **a** (append) command reads the last line of the file into memory, **Edlin** displays the message "End of input file".

Example:

Suppose you have a file so large that the last 100 lines would not fit into memory. After editing the first part of the file, and writing a portion back to the disk, you could use this command to read the remaining 100 lines into memory:

100a

For information about how to write edited lines to your disk, see the **w** (write) command in this chapter.

Edlin's Copy Command: c

Copying lines**Syntax:**

`[line],[line],line[,count]c`

Comments:

The **c** (copy) command copies a range of lines to a specified line number, and when used with the *count* option, copies this range as many times as you want. The first and second *line* options specify the range of lines that you want to copy. If you omit the first or second *line* option, **Edlin** defaults to the current line. The third *line* option specifies the line before which **Edlin** will place the copied lines.

You must not overlap the line numbers or you will get an "Entry error" message. For example, this command would result in an error message:

`3, 20, 15c`

If you do not specify a number for the *count* option, **Edlin** copies the lines one time and automatically rennumbers the file after the copy.

Example:

If you type

`1, 5, 6c`

Edlin copies lines 1 through 5 and duplicates them one time, beginning on line 6. Thus lines 1 through 5 and 6 through 10 are identical.

Deleting lines

Edlin's Delete Command: d**Syntax:**

*[line][,line]*d

Comments:

The **d** (delete) command deletes a specified range of lines in a file. If you omit the first *line* option, **Edlin** defaults to the current line (the line with the asterisk next to the line number). If you omit the second *line* option, **Edlin** deletes just the first *line*. Remember, too, that when you delete lines, **Edlin** automatically rennumbers the file.

Example:

If you type the following command, **Edlin** deletes line 7 and, then, rennumbers line 8 and all following lines:

```
7d
```

If you want to delete a block of text on lines 22 through 32, you can type

```
22, 32d
```

The **d** command removes lines 22 through 32, inclusively, from your file.

Finally, suppose that you want to delete a range of lines beginning with the current line, line 7, through line 11. Type the following:

```
, 11d
```

Edlin's Line Edit Command

Editing specified lines

Syntax:

[*line*]

Comments:

The *line* option allows you to specify the line number of text you want to edit. When you type a line number as a command, Edlin displays the line number and the text on that line; then, on the line below, Edlin reprints the line number. Now you can retype the line, or use the Edlin editing keys to edit it. The existing text of the line serves as the template until you press the ENTER key.

If you press the ENTER key without typing a line number, Edlin edits the line after the current line.

When you have edited the line, simply press the ENTER key to accept the line.

Warning

If you press the ENTER key while the cursor is in the middle of a line, Edlin deletes the remainder of the line.

Example:

Suppose that the following file exists and is ready to edit:

```
1: Dear Mr. Dimm,  
2:  
3: I was sorry to hear of your recent  
4: hospitalization due to electrical  
5: shock from our Automatic  
6: Pencil Sharpener.
```

In line 5, say you want to insert the product's name, X-1000. To edit line 5, type the number 5. Edlin then displays the contents of the line with the cursor below the line:

```
5:*shock from our Automatic  
5:*_
```

Now you simply use the **F2** key to skip to the **A** in the word *Automatic*, and type

F2 A INS X-1000

5:*shock from our X-1000

F3 ENTER

5:*shock from our X-1000 Automatic

At the **Edlin** prompt, type **L** to see the file:

1: Dear Mr. Dimm,

2:

3: I was sorry to hear of your recent
4: hospitalization due to electrical

5:*shock from our X-1000 Automatic

6: Pencil Sharpener.

Edlin's End/Save Command: e

Ending and saving edits

Syntax:

e

Comments:

The **e** (end) command saves the edited file on your disk, renames the original input file *filename.bak*, and then exits **Edlin**. If you created the file during this editing session, **Edlin** does not create a backup (*.bak*) file.

The **e** (end) command takes no options. This means that you must select the directory and drive that you want to save the file on *when you start Edlin*. If you don't select a drive when you start **Edlin**, it saves the file on the disk in the default drive. However, you can still copy the file to a different drive by using the MS-DOS **copy** command.

Before using the **e** command to save your file, make sure that the disk contains enough free space for the entire file. If it doesn't, **Edlin** may not be able to write the entire file to the disk. The edited file will be lost, although **Edlin** may have saved part of the file on the disk.

Notes:

If a *.bak* file already exists and is a read-only file, the updated file is saved by using the **e** (end) command. However, the original file does not detect this condition when you first start to edit a file, so MS-DOS displays this message:

```
Warning: Backup file is read only -  
backup will not be made.
```

Example:

To end an **Edlin** session and save the edits you have made, simply type

e

Edlin saves your edited file and returns you to the MS-DOS prompt (for example, A>).

Inserting lines of text

Edlin's Insert Command: i**Syntax:**

[*line*]i

Comments:

The *i* (insert) command allows you to insert text immediately before the specified *line*. If you are creating a new file, you must type the *i* (insert) command before you can insert a new line of text. Text begins on line 1, and the next line number appears automatically each time you press the ENTER key.

Edlin remains in insert mode until you press CONTROL+C. When you finish the insertion and exit insert mode, the line immediately following the inserted lines becomes the current line. **Edlin** automatically increments the line numbers that follow the inserted section by the number of lines that you inserted.

If you do not specify *line*, the default is the current line number and **Edlin** inserts the lines before the current line. If *line* is a number larger than the last line number, or if you specify a pound sign (#) as *line*, **Edlin** appends the inserted lines to the end of the file. In this case, the last line that you inserted becomes the current line.

Example:

Suppose the following file exists and is ready to edit:

```
1: Dear Mr. Dimm,  
2:  
3: I was sorry to hear of your recent  
4: hospitalization due to electrical  
5: shock from our X-1000 Automatic  
6: Pencil Sharpener.  
7:  
8: Sincerely,  
9:  
10: I.R. Sharpe, President
```

Inserting text in a file

This letter doesn't really offer any compensation for the accident, so you might want to add a comforting thought to the letter to console Mr. Dimm. To insert text before line 8, type *8i*. The result is

8: *_

Now type the following lines, which will begin on line 8:

```
8:*As a result of your accident, we
```

Press the ENTER key at the end of each line, and continue typing the next line:

```
9:*are redesigning our manual to
10:*warn our customers against trying
11:*to sharpen metal objects.
```

To end the insertion, press CONTROL+C on the *next* line.

To insert a blank line immediately before the current line (line 12), type *i*. The result is

```
12:*_
```

Insert a blank line by pressing ENTER, and end the insertion by pressing CONTROL+C on the next line. Then, to list the file and see the result, type *L*. The result is

```
1: Dear Mr. Dimm,
2:
3: I was sorry to hear of your recent
4: hospitalization due to electrical
5: shock from our X-1000 Automatic
6: Pencil Sharpener.
7:
8: As a result of your accident, we
9: are redesigning our manual to
10: warn our customers against trying
11: to sharpen metal objects.
12:
13:*Sincerely,
14:
15: I.R. Sharpe, President
```

**Inserting a blank line
in a file**

Listing a range of lines

Edlin's List Command: L**Syntax:**

[*line*][,*line*]L

Comments:

The **L** (list) command displays a range of lines, including the two lines specified. If you only specify one of the line options, **Edlin** uses default values. For example, if you omit the first *line* option, as in the following example, **Edlin** displays 23 lines, beginning 11 lines before the current line and ending with the specified *line*:

„*line* L

The beginning comma shows that you omitted the first *line* option. If you omit the second *line* option, **Edlin** displays 23 lines, starting with the specified *line*. If you type **L** with no *line* option at all, **Edlin** displays 23 lines—beginning with the 11 lines before the current line.

Notes:

- If the specified *line* is more than 11 lines before the current line, the display will be the same as if you omitted both options.
- A capital letter "L" has been used here to avoid confusion with the number "1" (one). A small letter "l" would work just as well.

Example:

To list lines 5 through 10, inclusive, type

5,10L

Edlin's Move Command: m

Moving specific lines

Syntax:

`[line,][+]line,linem`

Comments:

The **m** (move) command lets you transfer a block of text to another location in a file. The first and second *line* options specify the range of lines that you want to move. The third *line* option specifies the line to which you want to move the first line in the range.

Edlin automatically renumbers the lines after it moves them. For example, the following command moves the text from the current line—plus 25 lines—to line 100:

```
,+25,100m
```

If the line numbers that you specify overlap, **Edlin** displays an "Entry error" message.

Example:

Suppose the following file exists and is ready to edit.

```
1: Dear Mr. Dimm,
2:
3: I was sorry to hear of your recent
4: hospitalization due to electrical
5: shock from our X-1000 Automatic
6: Pencil Sharpener.
7:
8: As a result of your accident, we
9: are redesigning our manual to
10: warn our customers against trying
11: to sharpen metal objects.
12:
13: Sincerely,
14:
15: I.R. Sharpe, President
16: Sharpe Office Supplies
17: The World Leader in Office Sharpeware
18: Our motto: "You oughta be Sharpe too"
```

What if you prefer to have the motto at the start of the letter? If so, you could move lines 16-18 to line 1, by typing the following command:

```
16,18,1m
```

Moving a block of text in a file

The result of this command is

```
1: Sharpe Office Supplies
2: The World Leader in Office Sharpeware
3: Our motto: "You oughta be Sharpe too"
4: Dear Mr. Dimm,
5:
6: I was sorry to hear of your recent
7: hospitalization due to electrical
8: shock from our X-1000 Automatic
9: Pencil Sharpener.
10:
11: As a result of your accident, we
12: are redesigning our manual to
13: warn our customers against trying
14: to sharpen metal objects.
15:
16: Sincerely,
17:
18: I.R. Sharpe, President
```

Edlin's Paging Command: p

Paging

Syntax:

`[line][,line]p`

Comments:

The **p** (page) command displays a file one screen (23 lines) at a time. The first *line* option specifies the line at which **Edlin** starts displaying. The second *line* option specifies how many lines appear on each page. If you do not type the first *line*, **Edlin** starts the page at the line after the current line. If you do not type the second *line* option, **Edlin** lists 23 lines on each page.

Example:

To view lines 100 through 200 and to see the text one screen at a time, you could type this command:

`100,200p`

Quitting an editing session

Edlin's Quit/No Save Command: q**Syntax:****q****Comments:**

The **q** (quit) command is useful if you don't want to make any changes to a file. This command exits to the MS-DOS operating system and does *not* save any editing changes. If you use the **q** (quit) command, **Edlin** prompts you to make sure you don't want to save the changes. If you want to save changes as you exit **Edlin**, use the **e** (end) command.

Notes:

When you exit **Edlin**, it erases any previous copy of the file that has a *.bak* extension. But if you quit **Edlin** (**q**) and reply *Y* (for Yes) to the "Abort edit (Y/N)?" message, **Edlin** will not delete your previous backup copy.

Example:

The following example shows how to quit **Edlin** without saving your changes.

- 1 Press **CONTROL+C** to leave insert mode.
- 2 At the asterisk (*) prompt, type **q**.
- 3 The message "Abort edit (Y/N)?" appears.
- 4 Type *Y* (for Yes) to abort the edit, then press the **ENTER** key.

Edlin's Replace Command: r

Replacing text

Syntax:

```
[line][,line][?]rtext1 CONTROL+z text2
```

Comments:

The **r** (replace) command replaces all occurrences of a string of text in a range with a different string of text. The *line* options show the range that **r** (replace) uses. Each time Edlin finds *text1*, it replaces it with *text2*. Then Edlin displays each line that changes.

For example, the following command would change the word "mine" each time it occurred in a 20-line file to the word "ours":

```
1,20rmineCONTROL+zours
```

Notice that you press the **CONTROL** and **z** keys at the same time. Do not type "CONTROL+Z".

If a line contains two or more replacements, it is displayed once for each change. If you include a question mark (?) in your command, Edlin asks "O.K.?". If you press **Y** (for Yes) or **ENTER**, *text2* replaces *text1*, and Edlin looks for the next occurrence of *text1*. If you press any other key in response, Edlin does not make the change for that occurrence of *text1*. When Edlin has made all the changes, the asterisk prompt reappears.

Approving each replacement

When you do not specify *text1*, the **r** command assumes the old (any previous) value. If this is the first replacement that you have done during this editing session, and if you do not specify *text1*, the command ends. If you do not specify *text2*, you must end *text1* by pressing the **ENTER** key.

If you omit the first *line* option, Edlin uses the line after the current line, by default. The default for the second *line* option is the line following the last line of the file (represented by the symbol "#").

If you end *text1* with a **CONTROL+z** and do not specify *text2*, Edlin assumes you want blank spaces for *text2*. For example, suppose you want to delete all occurrences of the word "clients" from your file. To do this you could simply type the following command, then press **CONTROL+z** and **ENTER**:

```
rclients
```

The next command replaces *clients* with the previous *text2*:

```
rclients
```

The following command makes the previous *text1* become the previous *text2*:

```
r
```

Notice that *previous* refers to an earlier string of text specified in an *s* or *r* command.

Example:

Suppose the following file exists and is ready for editing:

```
1: Dear Mr. Dimm,
2:
3: I was sorry to hear of your recent
4: hospitalization due to electrical
5: shock from our X-1000 Automatic
6: Pencil Sharpener.
7:
8: As a result of your accident, we
9: are redesigning our manual to
10: warn our customers against trying
11: to sharpen metal objects.
12:
13: Sincerely,
14:
15: I.R. Sharpe, President
```

Now suppose that in lines 5 through 10 you want to replace all occurrences of the word *our* with the word *the*. To do this you would simply type **5,10 our; press CONTROL+Z, type *the*, and press the ENTER key.** The result is

```
5: shock from the X-1000 Automatic
8: As a result of ythe accident, we
9: are redesigning the manual to
10: warn the customers against trying
```

In the previous example, two unwanted changes occurred in lines 8 and 10. To avoid the changes and to confirm each replacement, you can use the same file with a slightly different command.

In the next example, you will see how to replace only certain occurrences of *our* with *the*. At the Edlin prompt, type the following sequence of keys and words, and then press the ENTER key:

Replacing text selectively

```
1,15? rour CONTROL+Z the
```

The result is

```
5: shock from the X-1000 Automatic
O.K.? y
8: As a result of ythe accident, we
O.K.? n
9: are redesigning the manual to
O.K.? y
10: warn the customers against trying
O.K.? n
*
_
```

Type the list command, L, to see the result of all these changes:

```
.
.
5: shock from the X-1000 Automatic
.
.
8: As a result of your accident, we
9: are redesigning the manual to
10: warn our customers against trying
.
.
```

Searching for text

Edlin's Search Command: s

Syntax:

`[line][,line][?]stext`

Comments:

The **s** (search) command searches a range of lines for a string of text. The first and second *line* options specify the range of lines for Edlin to search. You end the *text* option by pressing the ENTER key. Edlin displays the first line that matches the string; that line then becomes the current line. Unless you type the question mark (?) option, the **s** (search) command ends when it finds the first match. If Edlin cannot find a line with a match, it displays the message "Not found".

If you include the question mark option (?), Edlin displays the first line with matching text and prompts you with the message "O.K.?". If you press either *Y* (for Yes) or the ENTER key, this line becomes the current line and the search ends. If you press any other key, the search continues until another match is found, or until all lines have been searched. (The search ends when Edlin displays the "Not found message".)

If you do not type the first line number, Edlin defaults to the line *after* the current line; and if you do not type the second line number, it defaults to # (the line after the last line of the file).

If you omit the *text* option, Edlin uses the text from any previous **s** or **r** (replace) command. If this is the first **s** or **r** command you have used this session, and you have not specified a search string, the **s** command ends immediately.

Example:

Suppose the following file exists and is ready for editing:

```
1: Dear Mr. Dimm,  
2:  
3: I was sorry to hear of your recent  
4: hospitalization due to electrical  
5: shock from our X-1000 Automatic  
6: Pencil Sharpener.  
7:  
8: As a result of your accident, we  
9: are redesigning our manual to  
10: warn our customers against trying  
11: to sharpen metal objects.  
12:  
13: Sincerely,  
14:  
15: I.R. Sharpe, President
```

To search for the first occurrence of the word *to*, type the command **2,12 sto** and press the ENTER key. Edlin displays the following lines:

```
3: I was sorry to hear of your recent
```

To search through several occurrences of a string until the correct string is found, type the command **1, ? sto**. The result is

```
3: I was sorry to hear of your recent
O.K.?_
```

If you press any key (except *Y* or the ENTER key), the search continues, so type *N* (for No) here:

```
O.K.? n
```

Continue:

```
4: hospitalization due to electrical
O.K.?_
```

Now press *Y* to terminate the search:

```
O.K.? y
*_
```

Searching for a word in a file

**Transferring text from
another file****Edlin's Transfer Command: t****Syntax:**

[line]tfilename

Comments:

The **t** (transfer) command puts the contents of one file into another file, or into the text you are typing. **Edlin** inserts the *filename* at the line number you give in the *line* option, and then automatically renumbers the lines. If you omit the line number, **Edlin** inserts the text on the current line.

Example:

**Inserting one file into
another**

To copy a file named *irsharpe.mem* to line 12 of the file you are editing, use the following command:

```
12 t irsharpe.mem
```

Edlin's Write Command: w

Writing lines to disk

Syntax:

[*n*]w

Comments:

The **w** (write) command writes a specified number of lines to disk. The *n* option specifies the number of lines that you want to write to the disk. You need this command only if the file you are editing is too large to fit into memory. When you start Edlin, it reads lines from your file until memory is 3/4 full.

To edit the remainder of your file, you must write the edited lines in memory to your disk. Then you can load additional unedited lines from your disk into memory by using the **a** (append) command, which is described earlier in this chapter.

Example:

Suppose you had a file so big that the last 100 lines would not fit into memory. After you edited the first part of the file, you could free up enough space to edit the last part of the file with this command:

```
125w
```

Note

If you do not specify the number of lines for Edlin to write, it writes lines until memory is 3/4 full. But it does not write any lines to your disk until memory is more than 3/4 full. Also, Edlin rennumbers all of the lines so that the first remaining line becomes line number 1.

7 Link: A Linker

In this chapter, you'll learn

- How to create executable files with **link**
- How to use **link** command options
- How **link** creates programs

Note

You need to read this chapter only if you are writing programs in a computer language that creates object (*.obj*) files.

Introduction

The Microsoft 8086 Object Linker (**link**) creates executable programs from object files generated by the Microsoft Macro Assembler (MASM) or by compilers for high-level languages, such as C or Pascal. The linker copies the resulting program to an executable (*.exe*) output file. You can then run the program by typing the file's name on the MS-DOS command line.

To use **link**, you must create one or more object files, then submit these files, along with any required library files, to the linker for processing. **Link** combines code and data in the object files and searches the named libraries to resolve external references to routines and variables. It then copies a relocatable execution image and the relocation information to the executable file. Using the relocation information, MS-DOS can load the executable image at any convenient memory location and then run it. **Link** can process programs that contain up to one megabyte of code and data.

Starting and Using Link

How to start Link

This section explains three methods for starting and using the linker to create executable programs. These methods, which you can also mix, let you specify link files by

- Answering prompts
- Using a command line
- Using a response file

Once you start **link**, it will either process the files you supply or prompt you for additional files. Also, notice that you can stop the linker at any time by pressing **CONTROL+C**.

Method 1

Method 1: Using Prompts to Specify Link Files

When you type the command, **link**, at the MS-DOS prompt, the linker prompts you for the information it needs. Follow these steps:

- 1 First, type the following command and press the **ENTER** key:

```
link
```

Link prompts you for the object files you wish to link by displaying the following message:

```
Object Modules [.OBJ]:
```

- 2 Type the name or names of the object files you wish to link. If you do not supply extensions for these files, **link** supplies *.obj* by default. If you have more than one name, make sure you separate each with spaces or plus signs (+). If you have more names than can fit on one line, type a plus sign (+) as the last character on the line and press the **ENTER** key. **Link** then prompts you for additional object files.

Once you have given all your object filenames, press the **ENTER** key. The linker displays the following prompt:

```
Run File [filename.EXE]:
```

- 3 Notice that in step 2, *filename* is the same as the first *filename* you typed at the Object Modules prompt. Type the name of the executable file you wish to create, and press the **ENTER** key. If you do not give an extension, **link** supplies *.exe* by default.

If you want link to supply a default executable filename, just press the ENTER key. The *filename* will then be the same as the first object file, but with the extension *.exe*.

Once you have pressed the ENTER key, link displays the following prompt:

```
List File [NUL.MAP]:
```

- 4 Type the name of the map file you wish to create, then press the ENTER key. If you do not supply a filename extension, the linker uses *.map* by default. If you don't want a map file, don't type a filename. Just press the ENTER key.

Once you have pressed the ENTER key, link displays the following prompt:

```
Libraries [.LIB]:
```

- 5 Type the names of any library files containing routines or variables referenced but not defined in your program. If you give more than one name, make sure the names are separated by spaces or plus signs (+). If you don't supply filename extensions, the linker uses *.lib* by default. If you have more names than can fit on one line, type a plus sign (+) as the last character on the line and press the ENTER key. Link then prompts you for additional filenames.

After entering all the names, press the ENTER key. If you don't want to search any libraries, don't type any names; just press the ENTER key.

Link now creates the executable file.

When entering filenames, you must supply a pathname for any file that is not in the current drive and directory. You can use link options by typing them after the filename at any prompt. If the linker cannot find an object file, it displays a message and waits for you to change disks, if necessary.

At any prompt, you can type the rest of the filenames by using the command line format described in the next section, "Method 2: Using a Command Line to Specify Link Files".

You can also choose the default responses for all remaining prompts by typing a semicolon after any prompt, or you can type commas to indicate several files. (If you type a semicolon at the Object Modules prompt, be sure to supply at least one object filename.) When the linker encounters a semicolon, it immediately chooses the default responses and processes the remaining files without displaying any more prompts.

Using Link with prompts

Example:

The following example links the object modules *moda.obj*, *modb.obj*, *modc.obj*, and *startup.obj*; searches the library file *math.lib* on drive B of the *\lib* directory for routines and data used in the program; and creates an executable file named *moda.exe*, and a map file named *abc.map*. The */pause* option in the Object Modules prompt line then causes *link* to pause while you change disks, after which the linker creates the executable file (see the section entitled "Pausing to Change Disks", later in this chapter):

```
link
Object Modules [.OBJ]: moda+modb+
Object Modules [.OBJ]: modc+startup/PAUSE
Run File [moda.EXE]:
List File [NUL.MAP]: abc
Libraries [.LIB]: b:\lib\math
```

Method 2

Method 2: Using a Command Line to Specify Link Files

You can also create an executable program by typing *link*, followed by the names of the files you wish to process. The command line has the following general form:

```
link objectfiles [,executablefile] [,mapfile]
[,libraryfile]]] [options] [:]
```

The variables in this command line are described as follows:

objectfiles Includes the name or names of object files that you want to link together. The files must have been created using MASM or a high-level-language compiler. The linker requires at least one object file. If you do not supply an extension, *link* provides the extension *.obj*.

executablefile Is an optional placeholder for the name you wish to give the executable file that *link* will create. If you do not supply an *executablefile*, *link* creates a filename by using the name of the first object file in the command line and appending it with an *.exe* extension.

<i>mapfile</i>	Is the name of the file that receives the map listing. If you do not supply an extension, the linker provides the extension <i>.map</i> . If you specify the <i>/map</i> or <i>/linenumbers</i> option, the linker creates a map file even if you don't specify one in your command line.
<i>libraryfiles</i>	Includes the name or names of the libraries containing routines that you wish to link to create a program. If you do not supply an extension, link supplies the extension <i>.lib</i> .
<i>options</i>	Control the operation of link. You can use any of the options listed in the section entitled "The Link Options", later in this chapter. You can specify options anywhere on the command line.

The commas you use to separate filenames for the different types of files are required even if you don't supply a filename. If you want the filename for a file to be the default (the same as the base name of the first object file), you can type the comma that would follow the filename *without actually supplying the filename*. If you type the comma after the object file, link supplies the default name for the *executablefile* and suppresses the *mapfile* and *libraryfiles*. You can also use a semicolon anywhere after the object file to terminate the command line.

If you do not supply all filenames in the command line and do not end the command line with a semicolon, the linker prompts you for additional files, using the prompts described previously in the section, "Method 1: Using Prompts to Specify Link Files." If you give more than one object file or library file, you must separate the names by spaces or plus signs.

If you do not specify a drive or directory for a file, link assumes the file is on the current drive and directory. You cannot specify the drive or directory for the *objectfile* and expect link to supply the same drive and directory for other files. Instead, you must give the location of each file specifically.

Note

When linking modules produced by a high-level-language compiler that supports overlays, you must specify overlay modules by putting them in parentheses. Since MASM has no overlay manager, you can specify overlays only for object files linked with the run-time library of a language compiler that supports overlays.

For example, you can use overlays with modules compiled with Microsoft FORTRAN, versions 3.2 and later, Microsoft Pascal, versions 3.2 and later, and Microsoft C, versions 3.0 and later. See your language compiler's manual for details on specifying overlays.

Examples:

**Using Link with
command line options**

The first example below uses the object file *file.obj* to create the executable file *file.exe*. Link searches the *file.lib* library for routines and variables used within the program. It also creates a file called *file.map*, which contains a list of the program's segments and groups:

```
link file.obj,file.exe,file.map,routine.lib
```

The first example is equivalent to the following line:

```
link file , , , routine
```

The next example uses the two object files, *startup.obj* and *file.obj*, on the current drive to create an executable file named *file.exe* on drive B. Link creates a map file on the *\map* directory of the current drive, but does not search any libraries:

```
link startup+file,b:file,\map\file;
```

The final example links the object modules *moda.obj*, *modb.obj*, *modc.obj*, and *startup.obj*:

```
link moda modb modc startup/PAUSE, , abc,b:\lib\math
```

The linker searches through the library file *math.lib* in the *\lib* directory on drive B for routines and data used in the program. It then creates an executable file named *moda.exe*, and a map file named *abc.map*.

The */pause* option in the command line causes the linker to pause and ask you to change disks before it creates the executable file. (This option is described in more detail in the section, "Pausing to Change Disks," later in this chapter.)

Method 3: Using a Response File to Specify Link Files

Method 3

You can create a program by listing, in a response file, the names of all the files to be processed, and by giving the name of the response file on the link command line. The simplest way to use a response file is with a command line of the following form:

link @filename

You can also specify a response file at any prompt, or at any position in a command line. The input from the response file is treated exactly as though you had typed it at the link prompts or in a command line. However, any ENTER/LINEFEED combinations in the file are treated the same as if you had pressed the ENTER key in response to a prompt, or typed a comma in a command line.

When you specify a response file, remember that the filename must be the name of the response file, and that you must precede it by an “at” sign (@). If the file is in another directory or on another disk drive, you must provide a pathname.

You can name the response file anything you like. The file content has the following general form:

```
objectfiles  
[executablefile]  
[mapfile]  
[libraryfiles]
```

You can omit any elements that have already been provided at prompts or from within a partial command line.

You must place each group of filenames on a separate line. If you have more names than can fit on one line, you can simply continue the names on the next line by typing a plus sign as the last character in the current line and pressing the ENTER key. If you do not supply a filename for a group, you must leave an empty line. You can give options on any line.

You can place a semicolon on any line in the response file. When link encounters the semicolon, it automatically supplies default filenames for all files you have not yet named in the response file. The remainder of the response file is ignored.

When you create a program with a response file, the linker displays each response from your response file on the screen in the form of prompts. If the response file does not contain names for required files, link prompts you for the missing names and waits for you to enter responses.

The format of a response file

Note

A response file should end with either a semicolon (;) or an ENTER/LINEFEED combination. If you fail to provide a final ENTER/LINEFEED in the file, the linker will display the last line of the response file and wait for you to press the ENTER key.

Example:

Using Link with a response file

The following response file tells the linker to link the four object modules, *moda*, *modb*, *modc*, and *startup*. Then, before producing the executable file *moda.exe*, it tells link to pause to let you swap disks. Finally, the linker creates a map file *abc.map* and searches the *math.lib* library in the \lib directory of drive B:

```
moda modb modc startup /PAUSE
abc
b:\lib\math
```

The following procedure combines all three methods of supplying filenames. Assume you have a response file called *library* that contains one line:

```
lib1+lib2+lib3+lib4
```

Now start link with a partial command line:

```
link object1 object2
```

Link takes *object1.obj* and *object2.obj* as its object files, and prompts for the next file:

```
Run File [object1.EXE]: exec
List File [NUL.MAP]:
Libraries [LIB]: @library
```

You include the name *exec* so that the linker will name the executable file *exec.exe*. You then press the ENTER key to indicate that no map file is desired, and you enter *@library* so that the linker will read in the response file containing the four library filenames.

The Lib Files

You can direct **link** to search directories and disk drives for the libraries you have named in a command by either specifying one or more search paths with the library names, or by assigning the search paths to the environment variable **LIB** before you invoke **link**. Environment variables are explained under the **set** command in Chapter 3, “MS-DOS Commands.”

A search path is the path of a directory or drive name. You type search paths along with library names on the **link** command line or in response to the Libraries prompt. You can also specify up to 16 search paths and assign them to the **LIB** environment variable by using the MS-DOS **set** command. In the latter case, you must separate the search paths by semicolons (;).

If you include a drive or directory name in the filename for a library in the **link** command line, the linker searches there only. If you don't give a drive or directory name, **link** searches for library files in the following order:

- 1 First, the linker searches the current drive and directory.
- 2 If it doesn't find the library there and you have specified one or more search paths in the command line, **link** searches the specified search paths in the order in which you gave them.
- 3 If it still doesn't find the library and you have set a search path by using the **LIB** environment variable, the linker searches there.
- 4 If it still doesn't find the library, **link** prints an error message.

Examples:

In the first example, the linker searches only the `\altlib` directory on drive A to find the `math.lib` library. To find `common.lib`, it will search the current directory on the current drive, the current directory on drive B, and finally the `\lib` directory on drive D:

```
link file,,file,A:\altlib\math.lib+common+B:+D:\lib\
```

In the second example, **link** searches the current directory, the `\lib` directory on drive C, and the `\system\lib` directory on drive U to find the libraries, `math.lib` and `common.lib`:

```
set LIB=C:\lib;U:\system\lib
link file,,file.map,math+common
```

Using search paths with libraries

Searching for library files

Using Link with library files

The Map File

What is the map file?

The map file lists the names, load addresses, and lengths of all segments in a program. It also lists the names and load addresses of any groups in the program, the program start address, and messages about any errors it may have encountered. If the `/map` option is used in the `link` command line, the map file lists the names and load addresses of all public symbols.

Segment information has the general form shown in this example:

Start	Stop	Length	Name	Class
00000H	0172CH	0172DH	TEXT	CODE
01730H	01E19H	006EAH	DATA	DATA

The **Start** and **Stop** columns show the 20-bit addresses (in hexadecimal) of the first and last byte in each segment. These addresses are relative to the beginning of the load module, which is assumed to be address 0000H. The operating system chooses its own starting address once the program is actually loaded. The **Length** column gives the length of the segment in bytes; the **Name** column gives the name of the segment; and the **Class** column gives the segment's class name.

Group information has the following general form:

Origin	Group
0000:0	IGROUP
0173:0	DGROUP

In this example, **IGROUP** is the name of the code (instruction) group and **DGROUP** is the name of the data group.

At the end of the listing file, the linker gives you the address of the program entry point.

If you specify the `/map` option in the `link` command line, the linker adds a public-symbol list to the map file. The symbols are presented twice: once in alphabetical order, then in the order of their load addresses. The list has the general form shown in the following example:

Address	Publics by Name
0000:1567	BRK
0000:1696	CHMOD
0000:01DB	CHKSTK
0000:131C	CLEARERR
0173:0035	FAC

Address	Publics by Value
0000:01DB	CHKSTK
0000:131C	CLEARERR
0000:1567	BRK
0000:1696	CHMOD
0173:0035	FAC

The addresses of the public symbols are in segment:offset format. They show the location of the symbol relative to the beginning of the load module, which is assumed to be at address 0000:0000.

When the **/high** and **/dsallocate** options are used and the program's code and data combined do not exceed 64K bytes, the map file may show symbols that have unusually large segment addresses. These addresses indicate a symbol whose location is below the actual start of the program code and data.

For example, the following symbol entry shows that **TEMPLATE** is located below the start of the program:

```
FFF0:0A20          TEMPLATE
```

Notice that the 20-bit address of **TEMPLATE** is 00920H.

The Temporary Disk File—Vm.tmp

Link normally uses available memory for the link session. If it runs out of available memory, it creates a temporary disk file named *vm.tmp* in the current working directory. When the linker creates this file, it displays the following message:

```
VM.TMP has been created.
Do not change diskette in drive x:
```

After this message appears, you must not remove the disk from the drive specified by *x* until the link session ends. The **/pause** option cannot be used if a temporary file is created. After link has created the executable file, it deletes the temporary file automatically.

Note

Do not use the *vm.tmp* filename for your own files, since when the linker creates the temporary file, it destroys any previous file that has the same name.

The Link Options

Using Link options

The linker options specify and control the tasks that link performs. All options begin with the linker-option character, which is a slash (/). You can use the following options anywhere on a link command line:

Option name	Action
/help	Shows the list of options.
/pause	Pauses during linking.
/exepack	Packs executable files.
/map	Creates a public-symbol map.
/linenumbers	Copies line numbers to a map file.
/noignorecase	Preserves case sensitivity in names.
/nodefaultlibrarysearch	Overrides default libraries.
/stack	Sets maximum allocation space.
/cparmaxalloc	Sets the maximum number of 16-byte paragraphs needed by a program.
/high	Sets a high load address for a program.
/dsallocate	Allocates a data group.
/nogroupassociation	Sets a group association override.
/overlayinterrupt	Sets an overlay interrupt.
/segments	Sets a maximum number of segments.
/dosseg	Specifies MS-DOS segment ordering.

You can abbreviate an option name as long as your abbreviation contains enough letters to distinguish the specified option from other options. Minimum abbreviations are listed for each option.

Many of the link options set values in the MS-DOS program header. You will understand these options better if you understand how the header is organized. The program header is described in the *MS-DOS Programmer's Reference* and in some reference books on MS-DOS.

Viewing the Options List

The /help option

Syntax:

/help

The **/help** option causes **link** to write a list of the available options to the screen. If you ever need a reminder of the available options, you may find this list convenient. You should not give a filename when using the **/help** option.

Minimum abbreviation: **/he**

Example:

```
link /help
```

Pausing to Change Disks

The /pause option

Syntax:

/pause

The **/pause** option causes **link** to pause before writing the executable file to disk so that you can swap disks before the linker writes the executable (.exe) file to disk.

If you specify the **/pause** switch, the linker displays the following message before creating the run file:

```
About to generate .EXE file
Change diskette in drive x: and press <ENTER>
```

Notice that **x** is the proper drive name. This message appears after the linker has read data from the object files and library files, and after it has written data to the map file, if you specified one. **Link** resumes processing when you press the **ENTER** key. After it writes the executable file to disk, it displays the following message:

```
Please replace original diskette
in drive letter and press <ENTER>
```

Minimum abbreviation: **/p**

Note

Do not remove the disk used for the *vm.tmp* file, if such a file has been created. If the temporary disk message appears when you have specified the */pause* option, you should press **CONTROL+C** to terminate the link session. Rearrange your files so that the temporary file and the executable file can be written to the same disk, then try again.

Example:

The following command causes the linker to pause just before creating the executable file *file.exe*. After creating this file, link pauses again to let you replace the original disk:

```
link file/pause,file,,\lib\math
```

The /exepack option

Packing Executable Files**Syntax:**

/exepack

The **/exepack** option directs **link** to remove sequences of repeated bytes (typically nulls) and optimize the load-time relocation table before creating the executable file. Executable files linked with the **/exepack** option may be smaller, and, thus, load faster than files linked without the option. However, the Microsoft Symbolic Debug Utility (**symdeb**) cannot be used with packed files.

The **/exepack** option does not always save a significant amount of disk space (in some cases it may even increase file size). Programs that have a large number of load-time relocations (about 500 or more) and long streams of repeated characters will usually be shorter if packed. If you are not sure if your program meets these conditions, try linking it both ways and compare the results.

Minimum abbreviation: **/e**

Example:

This example creates a packed version of the file *program.exe*:

```
link program /e;
```

Producing a Public-Symbol Map

The /map option

Syntax:

/map

The **/map** option causes **link** to produce a listing of all public symbols declared in your program. This list is copied to the map file that **link** creates. For a complete description of the listing-file format, see the section, “The Map File,” earlier in this chapter. The **/map** option is required if you want to use **symdeb** for symbolic debugging.

Minimum abbreviation: **/m**

Note

If you do not specify a map file in a **link** command, you can use the **/map** option to force the linker to create one. **Link** gives the forced map file the same filename as the first object file specified in the command. It also adds the default extension **.map**.

Example:

The following command creates a map of all public symbols in the file **file.obj**:

```
link file,,/map;
```

Copying Line Numbers to the Map File

The /linenumbers option

Syntax:

/linenumbers

The **/linenumbers** option directs the linker to copy the starting address of each program source line to a map file. The starting address is actually the address of the first instruction that corresponds to the source line. You can use the **mapsym** program to copy line-number data to a symbol file, which can then be used by **symdeb**.

The linker copies the line-number data only if you give a map-file name in the **link** command line, and only if the given object file has line-number information. Line numbering is available in some high-level-language compilers, including Microsoft FORTRAN and Pascal, versions 3.0 and later, and Microsoft C, versions 2.0 and later.

MASM does not copy line-number information to the object file. If an object file has no line-number information, the linker ignores the `/linenumbers` option.

Minimum abbreviation: `/li`

Note

If you do not specify a map file in a **link** command, you can still use the `/linenumbers` option to force the linker to create one. Just place the option at or before the List File prompt. **Link** gives the forced map file the same filename as the first object file that you specified in the command, and gives it the default extension `.map`.

Example:

This example causes the line-number information in the object file *file.obj* to be copied to the map file *file.map*:

```
link file/linenumbers,,em+slibfp
```

The `/noignorecase` option

Preserving Lowercase

Syntax:

`/noignorecase`

The `/noignorecase` option directs **link** to treat uppercase and lowercase letters in symbol names as distinct letters. Normally, **link** considers uppercase and lowercase letters to be identical, treating the words "TWO", "two", and "Two" as the same symbol. When you use the `/noignorecase` option, however, the linker treats "TWO", "two", and "Two" as different symbols.

Typically, you use the `/noignorecase` option with object files created by high-level language compilers. Some compilers treat uppercase and lowercase letters as distinct letters and assume the linker does the same.

If you are linking modules created with MASM to modules created with a case-sensitive language such as C, make sure public symbols have the same sensitivity in both modules. For example, you could make all variables in C distinctive by spelling, regardless of case, and then link without the `/noignorecase` option. Another alternative would be to use the `/ML` or `/MX` option to make public variables in MASM case-sensitive. Then link with the `/noignorecase` option.

Minimum abbreviation: `/noi`

Example:

The following command causes the linker to treat uppercase and lowercase letters in symbol names as distinct letters. The object file *file.obj* is linked with routines from the standard C language library *\Slibc.lib* located in the *\lib* directory. The C language expects uppercase and lowercase letters to be treated distinctly:

```
link file1+file2/noi,,,em+mlibfp
```

Ignoring Default Libraries

The **/nodefault-librarysearch** option

Syntax:

/nodefaultlibrarysearch

The **/nodefaultlibrarysearch** option directs the linker to ignore any library names it may find in an object file. A high-level-language compiler may add a library name to an object file to ensure that a default set of libraries is linked with the program. Using this option overrides these default libraries and lets you explicitly name the libraries you want by including them on the link command line.

Minimum abbreviation: **/nod**

Example:

The following example links the object files, *startup.obj* and *file.obj*, with routines from the libraries, *em*, *slibfp*, and *slibc*. Any default libraries that may have been named in *startup.obj* or *file.obj* are ignored:

```
link startup+file/nod,,,em+slibfp+slibc
```

Setting the Stack Size

The **/stack** option

Syntax:

/stack:size

The **/stack** option sets the program stack to the number of bytes given by size. The linker usually calculates a program's stack size automatically, basing it on the size of any stack segments given in the object files. If you do use the **/stack** option, the linker uses the value you type in place of any value it may have calculated.

The size can be any positive integer in the range from 1 to 65,535. This value can be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero, and hexadecimal numbers must begin with a leading zero followed by a lowercase *x*, for example, *0x1B*.

By using the *exemod* utility, you can also change the stack size after linking.

Minimum abbreviation: */st*

Examples:

The first example sets the stack size to 512 bytes:

```
link file/stack:512,,;
```

The second example sets the stack size to 255 (FFH) bytes:

```
link moda+modb,run/st:0xFF,ab,\lib\start;
```

The final example sets the stack size to 24 (30 octal) bytes:

```
link startup+file/st:030,,;
```

The */cparmaxalloc* option

Setting the Maximum Allocation Space

Syntax:

/cparmaxalloc:number

The */cparmaxalloc* option sets the maximum number of 16-byte paragraphs needed by a program when it is loaded into memory. The operating system uses this number when allocating space for a program prior to loading it.

Link normally sets the maximum number of paragraphs to 65,535. Since this represents all addressable memory, the operating system always denies the default setting and allocates the largest contiguous block of memory it can find. If you use the */cparmaxalloc* option, the operating system allocates no more space than is given by this option. This means any additional space in memory is free for other programs.

The *number* can be any integer in the range from 1 to 65,535. It must be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero, and hexadecimal values must begin with a leading zero followed by a lowercase *x*, for example, *0x2B*.

If *number* is less than the minimum number of paragraphs needed by the program, **link** ignores your request and sets the maximum value equal to the minimum needed. The minimum number of paragraphs needed by a program is never less than the number of paragraphs of code and data in the program.

Minimum abbreviation: `/c`

Examples:

The first example sets the maximum allocation to 15 paragraphs:

```
link file/c:15,,;
```

The second example sets the maximum allocation to 255 (FFH) paragraphs:

```
link moda+modb,run/cparmaxalloc:0xff,ab;
```

The final example sets the maximum allocation to 24 (30 octal) paragraphs:

```
link startup+file,/c:030,;
```

Setting a High Start Address

The `/high` option

Syntax:

`/high`

The **`/high`** option sets a program's starting address to the highest possible address in free memory. If you don't use the **`/high`** option, **link** sets the program's starting address as low as possible in memory.

Minimum abbreviation: `/h`

Example:

This example sets the starting address of the program in *file.exe* to the highest possible address in free memory:

```
link startup+file/high,,;
```

The /dsallocate option

Allocating a Data Group**Syntax:****/dsallocate**

The **/dsallocate** option directs the linker to reverse its normal processing when assigning addresses to items belonging to the group named DGROUP. Normally, **link** assigns the offset 0000H to the lowest byte in a group. If you use **/dsallocate**, **link** assigns the offset FFFFH to the highest byte in the group. The result is data that appear to be loaded as high as possible in the memory segment containing DGROUP.

Typically, you use the **/dsallocate** option with the **/high** option to take advantage of unused memory before the start of the program. The linker assumes that all free bytes in DGROUP occupy the memory preceding the program. To use the group, you must set a segment register to the start address of DGROUP.

Minimum abbreviation: **/d**

Example:

The following example directs the linker to place the program as high in memory as possible, then adjust the offsets of all data items in DGROUP so that they are loaded as high as possible within the group:

```
link startup+file/high/dsallocate,,,em+m1ibfp
```

The /nogroup- association option

Removing Groups from a Program**Syntax:****/nogroupassociation**

The **/nogroupassociation** option directs **link** to ignore group associations when assigning addresses to data and code items.

Minimum abbreviation: **/nog**

Note This option exists strictly for compatibility with older versions of FORTRAN and Pascal (Microsoft versions 3.13 or earlier, or any IBM version prior to 2.0). You should never use the **/nogroupassociation** option except to link with object files produced by those compilers, or with the run-time libraries that accompany the old compilers.

Setting the Overlay Interrupt

The `/overlay-`Interrupt option

Syntax:

`/overlayinterrupt:number`

The `/overlayinterrupt` option sets the interrupt number of the overlay loading routine to *number*. This option overrides the normal overlay interrupt number (03FH).

Number can be any integer value in the range from 0 to 255. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero, and hexadecimal numbers must start with a leading zero followed by a lowercase *x*, for example, *0x3B*.

MASM does not have an overlay manager. Therefore, you can use this option only if you are linking with a run-time module from a language compiler that supports overlays. Check your compiler documentation, since you may not be able to use this option with some compilers.

Minimum abbreviation: `/o`

Note

You should not use interrupt numbers that conflict with the standard MS-DOS interrupts.

Examples:

The first example sets the overlay interrupt number to 255:

```
link file/o:255,,,87+slibfp
```

The second example sets the overlay interrupt number to 255 (FFH):

```
link moda+modb, run/overlay:0xff,ab.map,em+mlibfp
```

The final example sets the overlay interrupt number to 255 (377 octal):

```
link startup+file,/o:0377,,em+mlibfp
```

The /segments option

Setting the Maximum Number of Segments**Syntax:**

/segments:number

The */segments* option directs the linker to process no more than *number* segments per program. If it encounters more than the given limit, the linker displays an error message, and stops linking. You use this option to override the default limit of 128 segments.

If you do not use */segments*, the linker allocates enough memory space to process up to 128 segments. If your program has more than 128 segments, you will need to set the segment limit higher to increase the number of segments that *link* can process. If you get the following *link* error message, you should set the segment limit lower:

```
Segment limit set too high
```

The *number* can be any integer value in the range from 1 to 1024. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero, and hexadecimal numbers must start with a leading zero followed by a lowercase *x*, for example, *0x4B*.

Minimum abbreviation: */se*

Examples:

The first example sets the segment limit to 192:

```
link file/se:192,,;
```

The second example sets the segment limit to 255 (FFH).

```
link moda+modb,run/segments:0xff,ab,em+mllibfp;
```

The /dosseg option

Using DOS Segment Order**Syntax:**

/dosseg

The */dosseg* option causes *link* to arrange all segments in the executable file according to the MS-DOS segment-ordering convention. This convention has the following rules:

- All segments having the class name `CODE` are placed at the beginning of the executable file.
- Any other segments that do not belong to the group, `DGROUP`, are placed immediately after the `CODE` segments.
- All segments belonging to `DGROUP` are placed at the end of the file.

If you do not use the `/dosseg` option, see the section, “Order of Segments”, later in this chapter, for an explanation of the normal segment order.

Minimum abbreviation: `/do`

Example:

The following command causes the linker to create an executable file, named *file.exe*, whose segments are arranged according to the MS-DOS segment-ordering convention. The segments in the object files *start.obj* and *test.obj*, and any segments copied from the libraries *math.lib* and *common.lib*, are arranged according to the same segment-ordering convention as above.

```
link start+test/dosseg,,math+common
```

How Link Works

Link creates an executable file by concatenating a program's code and data segments according to the instructions in the original source files. These concatenated segments form an *executable image* that is copied directly into memory when you run the program. The order and manner in which the linker copies segments to the executable file defines the order and manner in which it loads the segments into memory.

You can tell the linker how to link a program's segments by using a `SEGMENT` directive to supply segment attributes, or by using the `GROUP` directive to form segment groups. These directives define group associations and classes, and align and combine types that define the order and relative starting addresses of all segments in a program. This information works in addition to any information you supply through command line options.

The following sections explain the process that **link** uses to concatenate segments and resolve references to items in memory.

Alignment of Segments

How link aligns segments

The linker uses a segment's align type to set the starting address for the segment. The align types are *byte*, *word*, *para*, and *page*. These types correspond to starting addresses at byte, word, paragraph, and page boundaries, representing addresses that are multiples of 1, 2, 16, and 256, respectively. The default align type is *para*.

When the linker encounters a segment, it checks the align type before copying the segment to the executable file. If the align type is *word*, *para*, or *page*, the linker checks the executable image to see if the last byte copied ends at an appropriate boundary. If it doesn't, link pads the image with extra null bytes.

Frame Number

Segment starting addresses

The linker computes a starting address for each segment in a program. The starting address is based on a segment's align type and on the size of the segments already copied to the executable file. The address consists of an offset and a canonical frame number, which specifies the address of the first paragraph in memory that contains one or more bytes of the segment. A frame number is always a multiple of 16 (a paragraph address), and the offset is the number of bytes from the start of the paragraph to the first byte in the segment. For *byte* and *word* align types, the offset may be nonzero, but the offset is always zero for *para* and *page* align types.

The frame number of a segment can be obtained from a link file. The frame number is the first five hexadecimal digits of the start address specified for the segment.

Order of Segments

Segment order

Link copies segments to the executable file in the same order that it encounters them in the object files. The linker maintains this order throughout the program unless it encounters two or more segments with the same class name. Segments with identical class names belong to the same class type, and are ordered contiguously.

The *Microsoft Macro Assembler Reference Manual* includes a more detailed discussion of segment loading order and methods of controlling loading order by assigning class types.

Combined Segments

Link uses combine types to determine whether two or more segments sharing the same name should be combined into a single large segment. The combine types are *public*, *stack*, *common*, *memory*, *at*, and *private*. Combine types are also described in the *Microsoft Macro Assembler Reference Manual*.

If a segment has a *public* combine type, the linker automatically combines it with any other segments that have the same name and belong to the same class. When **link** combines segments, it ensures that the segments are contiguous and that all addresses in the segments can be accessed using an offset from the same frame address. The result is the same as if the segment were defined as a whole in the source file.

The linker preserves each segment's align type. This means that even though the segments belong to a single, large segment, the code and data in the segments retain their original align type. If the combined segments exceed 64K bytes, **link** displays an error message.

If a segment has a *stack* combine type, the linker carries out the same combine operation as for public segments. The only difference is that stack segments cause **link** to copy an initial stack-pointer value to the executable file. This stack-pointer value is the offset to the end of the first stack segment (or combined stack segment) that the linker encounters.

If you use the *stack* type for stack segments, you do not need to give instructions to load the segment into the SS register.

If a segment has a *common* combine type, the linker combines it automatically with any other segments that have the same name and belong to the same class. When **link** combines common segments, however, it places the start of each segment at the same address, creating a series of overlapping segments. The result is a single segment no larger than the largest of the combined segments.

The linker treats segments with *memory* combine types exactly like segments with *public* combine types. The Microsoft Macro Assembler (MASM) provides combine type memory for compatibility with linkers that support a separate combine type for memory segments.

An *at* segment typically contains no code or initialized data. Instead, it represents an address template that can be placed over code or data already in memory, such as a screen buffer or other absolute memory locations defined by hardware. The linker will not generate any code or

Combine types

data for *at* segments, but existing code or data can be accessed by name if it is given a label in an *at* segment.

A segment has a *private* combine type only if no explicit combine type is defined for it in the source file. **Link** does not combine private segments.

Groups

Groups

Groups permit noncontiguous segments that do not belong to the same class to be addressable relative to the same frame address. When **link** encounters a group, it adjusts all memory references to items in the group so that they are relative to the same frame address.

Segments in a group do not have to be contiguous and do not have to belong to the same class. Nor do they have to have the same combine type. The only requirement is that all segments in the group fit within 64K bytes.

Groups do not affect the order in which the segments are loaded. Unless you use class names and enter object files in the right order, there is no guarantee that the segments will be contiguous. In fact, the linker may place segments that do not belong to the group in the same 64K bytes of memory. Although **link** does not explicitly check whether all segments in a group fit within this 64K of memory, the linker is likely to encounter a *fixup-overflow* error if this requirement is not met.

Groups, and how to define them, are discussed further in the *Microsoft Macro Assembler Reference Manual*.

Fixups

Once the starting address of each segment in a program is known, and all segment combinations and groups have been established, the linker can fix up any unresolved references to labels and variables. To fix up unresolved references, the linker computes an appropriate offset and segment address and replaces the temporary values, generated by the assembler, with the new values.

Reference fixups

Link carries out fixups for four different references:

- Short
- Near self-relative
- Near segment-relative
- Long

The size of the value to be computed depends on the type of reference. If **link** discovers an error in the anticipated size of a reference, it displays a *fixup-overflow* message. This error can happen, for example, if a program attempts, by using a 16-bit offset, to reach an instruction in a segment that has a different frame address. The error can also occur if the segments in a group do not fit within a single 64K-byte block of memory.

A *short reference* occurs in JMP instructions that attempt to pass control to labeled instructions in the same segment or group. The target instruction must be no more than 128 bytes from the point of reference. The linker computes a signed, 8-bit number for this reference and displays an error message if the target instruction belongs to a different segment or group (that is, if it has a different frame address), or if the target is more than 128 bytes distant (in either direction).

A *near self-relative reference* occurs in instructions which access data relative to the same segment or group. The linker computes a 16-bit offset for this reference and displays an error message if the data are not in the same segment or group.

A *near segment-relative reference* occurs in instructions that attempt to access data in a specified segment or group, or data that are relative to a specified segment register. **Link** computes a 16-bit offset for this reference and displays an error message if either of the following conditions exists: the offset of the target within the specified frame is greater than 64K bytes or less than 0, or the beginning of the canonical frame of the target is not addressable.

A *long reference* occurs in CALL instructions that attempt to access an instruction in another segment or group. **Link** computes a 16-bit frame address and 16-bit offset for this reference and displays an error message if either of the following conditions exists: the computed offset is greater than 64K bytes or less than 0, or the beginning of the canonical frame of the target is not addressable.

In this chapter, you've learned how to use **link** to create MS-DOS executable files. You've also learned how to use **link** command options, and some details about how **link** works. In the next chapter, "Debug," you'll learn how to use the **debug** utility when writing or testing binary (.com or .exe) executable object files.

What comes next?

8 Debug

In this chapter, you will learn

- How to start the **debug** utility
- How to use the **debug** commands and parameters

Note

You need to read this chapter only if you are writing or testing executable programs (*.com* or *.exe* files).

Introduction

The **debug** utility is a debugging program that provides a controlled testing environment for binary and executable object files. Notice that **Edlin**, the MS-DOS line editor, is used to alter source files; **debug** is **Edlin**'s counterpart for binary files.

Debug eliminates the need to reassemble a program to see if a problem has been fixed by a minor change. It allows you to alter the contents of a file or the contents of a CPU register, and then immediately reexecute a program to check the validity of the changes made.

All **debug** commands may be aborted at any time by pressing **CONTROL+C**. The **CONTROL+S** key sequence suspends the display, so that you can read it before the output scrolls away. Pressing any key other than **CONTROL+C** or **CONTROL+S** restarts the display. All these commands are consistent with the control character functions available at the MS-DOS command level.

How to Start Debug

Starting Debug

Debug may be started two ways. By the first method, you type all commands in response to the **debug** prompt (a hyphen). By the second method, you type all commands on the line used to start **debug**.

Method 1: **debug**

Method 2: **debug** *[filename [arglist]]*

Method 1

Method 1: Debug

To start **debug** using method 1, simply type the following:

```
debug
```

Debug responds with the hyphen (-) prompt, signaling that it is ready to accept your commands. Since you didn't specify a filename, you can use other commands to work on current memory, disk sectors, or disk files.

Warnings

- When **debug** (version 3.0) is started, it sets up a program header at offset 0 in the program work area. In previous versions of **debug**, you could overwrite this header. You can still overwrite the default header if you don't give a *filename* to **debug**. If you are debugging a *.com* or *.exe* file, however, do not tamper with the program header below address 5CH, or **debug** will terminate.
- Do not restart a program after the following message is displayed:

```
Program terminated normally
```

You must reload the program with the **N** (name) and **L** (load) commands for it to run properly.

Method 2: Command Line

Method 2

To start **debug** using a command line, you must use the following syntax:

debug [*filename* [*arglist*]]

For example, if you specify a filename, the following would be atypical command to start **debug** :

```
debug file.exe
```

Debug would then load *file.exe* into memory starting at 100 (hexadecimal) in the lowest available segment. The **BX:CX** registers are loaded with the number of bytes placed into memory.

If you do include a *filename*, you might also specify an *arglist*. An *arglist* is a list of filename parameters and switches that are to be passed to the program *filename*. So when *filename* is loaded into memory, it is loaded as if it had been started with a command of the form, **debug filename arglist**.

Here, *filename* is the file to be debugged, and *arglist* is the rest of the command line used when **debug** calls and loads filename into memory.

Debug Command Information

Each **debug** command consists of a single letter followed by one or more parameters. Additionally, the control characters and special editing functions described in Chapter 5, “MS-DOS Editing and Function Keys”, apply to **debug** as well.

If a syntax error occurs in a **debug** command, **debug** reprints the command line and indicates the error with a caret (^) and the word “Error” as in the following example,

```
dcs:100 cs:110
^ Error
```

Notice that when typing commands and parameters you may use any combination of uppercase and lowercase letters.

About Debug commands

The debug commands are listed below. Following this list, the commands and their parameters are described in greater detail.

Debug command	Function
A [<i>address</i>]	Assemble
C <i>range address</i>	Compare
D [<i>range</i>]	Dump
E <i>address [list]</i>	Enter
F <i>range list</i>	Fill
G [= <i>address</i> [<i>address...</i>]]	Go
H <i>value value</i>	Hex
I <i>value</i>	Input
L [<i>address</i> [<i>drive:record record</i>]]	Load
M <i>range address</i>	Move
N <i>filename [filename]</i>	Name
O <i>value byte</i>	Output
P [= <i>address</i>][<i>number</i>]	Proceed through loop or subroutine
Q	Quit
R [<i>register-name</i>]	Register
S <i>range list</i>	Search
T [= <i>address</i>] [<i>value</i>]	Trace
U [<i>range</i>]	Unassemble
W [<i>address</i> [<i>drive:record record</i>]]	Write
XA	Expanded memory allocate
XD	Expanded memory deallocate
XM	Expanded memory map pages
XS	Expanded memory status

Debug Command Parameters

All **debug** commands accept parameters, except the **Q** (quit) command. Parameters may be separated by delimiters (spaces or commas), but a delimiter is required only between two consecutive hexadecimal values. Thus, the following commands are equivalent:

```
dcs:100 110
d cs:100 110
d,cs:100,110
```

How to use the Debug command parameters

Parameter	Definition
<i>drive:</i>	A one-digit hexadecimal value that indicates which drive a file will be loaded from or written to. The valid values are 03, where 0=A:, 1=B:, 2=C:, 3=D:.
<i>byte</i>	A two-digit hexadecimal value placed in or read from an address or register.
<i>record</i>	A one-digit to three-digit hexadecimal value that indicates the logical record number on the disk and the number of disk sectors to be written or loaded. Logical records correspond to sectors; however, since they represent the entire disk space, their numbering differs.
<i>value</i>	A hexadecimal value of up to four digits that specifies a port number or the number of times a command should repeat its functions.
<i>address</i>	A two-part designation containing either an alphabetic segment register or a four-digit segment address plus an offset value. You may omit the segment name or segment address, in which case the default segment DS is used for all commands except A, G, L, T, U, and W, for which the default segment is CS. All numeric values are hexadecimal.

Following is a sample *address*:

```
CS:0100
04BA:0100
```

Notice that the colon is required between the segment name (whether numeric or alphabetic) and the offset value.

Parameter	Definition
<i>range</i>	Contains two addresses: for example, <i>address address</i> ; or one address, an L, and a value: for example, <i>address L value</i> where <i>value</i> is the number of lines on which the command should operate (L80 is assumed). The second type of <i>range</i> cannot be used if another hexadecimal value follows, since the hexadecimal value would be interpreted as the second <i>address of the range</i> .

Here are some sample ranges:

```
CS:100 110
CS:100 L 10
CS:100
```

The following example, however, is illegal:

```
CS:100 CS:110
    ^ Error
```

The limit for *range* is 10000 (hexadecimal). To specify a *value* of 10000 with only four digits, type 0000 (or 0).

list A series of byte values or *strings*. *List* must be the last parameter on the command line.

Following is a sample *list*:

```
fcs:100 42 45 52 54 41
```

string Any number of characters enclosed in quotation marks. The quotation marks may be either single (' ') or double (" "). If the delimiter marks must appear within a *string*, you must use the double quotation marks.

Parameter	Definition
	For example, the following strings are legal: <pre>"This 'string' is okay." "This ""string"" is okay."</pre> However, this string is illegal: <pre>""This "string" is not okay.""</pre> Notice that the double quotation marks are not necessary in the following strings: <pre>"This 'string' is not necessary." 'This ""string"" is not necessary.'</pre> The <code>ASCII</code> values of the characters in the string are used as a <i>list</i> of byte values.
<i>count</i>	Number of 16K pages to allocate.
<i>handle</i>	EMS handle number.
<i>lpage</i>	Logical page of handle to map.
<i>ppage</i>	Physical page where logical page will map into.

(A)semble

Assemble**Purpose:**

Assembles 8086/8087/8088 mnemonics directly into memory.

Syntax:

A[address]

Comments:

If it finds a syntax error, **debug** responds with the following message, then redisplay the current assembly address:

^Error

All numeric values are hexadecimal and you must type them as 14 characters. Also, you must specify prefix mnemonics *in front of* the opcode to which they refer. You may type them on a separate line, however.

The segment override mnemonics are CS:, DS:, ES:, and SS:. The mnemonic for the far return is **RETF**. String manipulation mnemonics must explicitly state the string size. For example, use **MOVSW** to move word strings, and use **MOVSB** to move byte strings.

The assembler will automatically assemble short, near, or far jumps and calls, depending on byte displacement, to the destination address. You may override these jumps and calls by using a **NEAR** or **FAR** prefix, as in the following example:

```
0100:0500 JMP 502 ; a 2-byte short jump
0100:0502 JMP NEAR 505 ; a 3-byte near jump
0100:505 JMP FAR 50A ; a 5-byte far jump
```

You may abbreviate the **NEAR** prefix to **NE**, but the **FAR** prefix cannot be abbreviated.

Debug cannot tell whether some operands refer to a word memory location or to a byte memory location. In this case, the data type must be explicitly stated with the prefix, **WORD PTR** or **BYTE PTR**. Acceptable abbreviations are **WO** and **BY**. For example,

```
NEG BYTE PTR [128]
DEC WO [SI]
```

Debug also cannot tell whether an operand refers to a memory location or to an immediate operand. So it uses the common convention that operands enclosed in square brackets refer to memory. For example,

```
MOV      AX, 21 ;      Load AX with 21H
MOV      AX, [21];      Load AX with the
                        ; contents
                        ; of memory location 21H
```

Two popular pseudo-instructions are available with the **A** (assemble) command: the **DB** opcode, which assembles byte values directly into memory; and the **DW** opcode, which assembles word values directly into memory. Following are examples of both:

```
DB      1,2,3,4,"THIS IS AN EXAMPLE"
DB      'THIS IS A QUOTATION MARK: "'
DB      "THIS IS A QUOTATION MARK: '"

DW      1000,2000,3000,"BACH"
```

The **A** command supports all forms of register indirect commands. For example,

```
ADD      BX, 34[BX+2] . [SI-1]
POP      [BP+DI]
PUSH     [SI]
```

All opcode synonyms are also supported, as in the next example:

```
LOOPZ    100
LOOPE    100

JA       200
JNBE     200
```

For 8087 opcodes, the **WAIT** or **FWAIT** prefixes must be explicitly specified, as in the last example:

```
FWAIT    FADD  ST,ST(3) ; This line assembles
                        ; an FWAIT prefix
```

(C)ompare

Compare**Purpose:**

Compares the portion of memory specified by *range* to a portion of the same *size* beginning at the specified *address*.

Syntax:

C range address

Comments:

If the two areas of memory are identical, there is no display, and **debug** returns with the MS-DOS prompt. If there *are* differences, they are displayed in this format:

address1 byte1 byte2 address2

Example:

Comparing blocks of memory

The following commands have the same effect:

```
C100,1FF 300
```

or:

```
C100L100 300
```

Each command compares the block of memory from 100 to 1FFH with the block of memory from 300 to 3FFH.

Dump

(D)ump

Purpose:

Displays the contents of the specified range of memory.

Syntax:

D[*range*]

Comments:

If you specify a *range* of addresses with the **D** (dump) command, the contents of the *range* are displayed. If you don't use parameters with the **D** command, 128 bytes are displayed at the first address (DS:100) after the address displayed by the previous **D** command.

The dump is displayed in two portions: a hexadecimal dump (each byte is shown in hexadecimal value) and an ASCII dump (the bytes are shown in ASCII characters). Nonprinting characters are denoted by a period (.) in the ASCII portion of the display. Each display line shows 16 bytes, with a hyphen between the eighth and ninth bytes. Each displayed line begins on a 16-byte boundary. At times in this chapter, displays are split to fit them on the page.

Examples:

If you type the command

```
dcx:100 110
```

debug displays the dump in the following format:

```
04BA:0100 42 45 52 54 41 ... 4E 44 TOM SAWYER
```

If you simply type the **D** command, the display is formatted as just described. Each line of the display begins with an address incremented by 16 from the address on the previous line.

Each subsequent **D** (typed without parameters) displays the bytes immediately following those last displayed.

Using the Dump command

If you type the following command, the display is formatted as described above, but 20H bytes are displayed:

```
DCS:100 L 20
```

If you then type the following command, the display is formatted as described above, but all the bytes in the range of lines from 100H to 115H in the CS segment are displayed:

```
DCS:100 115
```



Enter

(E)nter

Purpose:

Enters byte values into memory at the specified *address*.

Syntax:

Eaddress[list]

Comments:

If, when using the E (enter) command, you type the optional *list* of values, the byte values are replaced automatically. (If an error occurs, no byte values are changed.)

If you type the *address* without the optional *lists*, **debug** displays the *address* and its contents, repeats the *address* on the next line, and then waits for your input. At this point, the E (enter) command waits for you to perform one of the following actions:

- Replace a byte value with a value you type. Simply type the value after the current value. If the one you type is not a legal hexadecimal value or if it contains more than two digits, the illegal or extra character is not echoed.
- Press the SPACEBAR to advance to the next byte. To change the value, simply type the new value as described in the previous action. If, when you press the SPACEBAR, you move beyond an 8-byte boundary, **debug** starts a new display line with the address displayed at the beginning.
- Type a hyphen (-) to return to the preceding byte. If you decide to change a byte behind the current position, typing the hyphen returns the current position to the previous byte. When you type the hyphen, a new line is started with its address and byte value displayed.
- Press the ENTER key to terminate the E command. The ENTER key may be pressed at any byte position.

Using the Enter command

Examples:

Using the Enter command

Suppose you type the following command:

```
ECS:100
```

Now suppose that **debug** displays the following:

```
04BA:0100 EB. __
```

To change this value to, say, 41, type the number 41 at the cursor, as shown:

```
04BA:0100 EB. 41__
```

To step through the subsequent bytes, you would press the **SPACEBAR** until you saw the following:

```
04BA:0100 EB. 41 10. 00. BC. __
```

To change **BC** to the number 42, for instance, you would type the number at the cursor, as follows:

```
04BA:0100 EB. 41 10. 00. BC. 42__
```

Notice that the value 10 should be 6F. To correct this value, you would type the hyphen as many times as needed to return to byte 0101 (value 10), then replace 10 with 6F:

```
04BA:0100 EB. 41 10. 00. BC. 42-
```

```
04BA:0102 00. -__
```

```
04BA:0101 10. 6F__
```

Pressing the **ENTER** key ends the **E** command and returns you to the **debug** command level.

Fill

(F)III

Purpose:

Fills the addresses in the specified *range* with the values in the specified *list*.

Syntax:

*Fr*ange *list*

Comments:

If the range contains more bytes than the number of values in the *lists*, the *lists* will be used repeatedly until all bytes in the *range* are filled.

If the *lists* contains more values than the number of bytes in the *range*, the extra values in the *lists* are ignored. If any of the memory in the *range* is not valid (bad or nonexistent), the error will occur in all succeeding locations.

Example:

Suppose you type the following command:

```
F04BA:100 L 100 42 45 52 54 41
```

Using the Fill command

In response, **debug** would fill memory locations 04BA:100 through 04BA:1FF with the bytes specified. The five values would then be repeated until all the 100H bytes were filled.

(G)o

Go**Purpose:**

Executes the program currently in memory.

Syntax:

G[=*address* [*addresses*]]

Comments:

If you type the **G** (go) command by itself, the program currently in memory executes as if it had run outside **debug**.

If you set **=*address***, execution of the **G** command begins at the address specified. The equal sign (=) is required so that **debug** can distinguish the start **=*address*** from the breakpoint *addresses*.

With the other optional *addresses* set, execution stops at the first *address* encountered, regardless of that address' position in the list of addresses that halt execution or program branching. When program execution reaches a breakpoint, the registers, flags, and decoded instruction are displayed for the last instruction executed. The result is the same as if you had typed the **R** (register) command for the breakpoint address.

You may set up to ten breakpoints, but only at addresses containing the first byte of an 8086 opcode. If you set more than ten breakpoints, **debug** returns the **BP** error message.

The user stack pointer must be valid and must have 6 bytes available for this command. The **G** command uses an **IRET** instruction to cause a jump to the program under test. The user stack pointer is set, and the user flags, Code Segment register, and Instruction Pointer are pushed on the user stack. (If the user stack is not valid or is too small, the operating system may crash.) An interrupt code (0CCH) is placed at the specified breakpoint address(es).

When **debug** encounters an instruction with the breakpoint code, it restores all breakpoint addresses to their original instructions. If you don't halt execution at one of the breakpoints, the interrupt codes are not replaced with the original instructions.

Example:

Using the Go command

Suppose you type the following command:

```
GCS:7550
```

The program currently in memory would execute up to the address 7550 in the CS segment. **Debug** would then display registers and flags, after which the G command would terminate.

After **debug** has encountered a breakpoint, if you type the G command again the program runs as if you had typed the filename at the MS-DOS command level. The only difference is that program execution begins at the instruction after the breakpoint, rather than at the usual start address.

(H)ex

Hex

Purpose:

Performs hexadecimal arithmetic on the two specified parameters.

Syntax:

Hvalue value

Comments:

First, **debug** adds the two parameters, then subtracts the second parameter from the first. The results of these actions are displayed on one line—first the sum, then the difference.

Example:

Using the Hex command

Suppose you type the following command:

```
H19F 10A
```

In response, **debug** would perform the calculations and then display the following result:

```
02A9 0095
```

Input

(I)input

Purpose:

Input and displays one byte from the port specified by value.

Syntax:

Ivalue

Comment:

A 16-bit port address is allowed.

Example:

Suppose you type the following command:

```
I2F8
```

Suppose also that the byte at the port is 42H. **Debug** would input the byte and then display the following:

```
42
```

Using the Input command

(L)oad

Load**Purpose:**

Loads a file into memory.

Syntax:

L[*address* [*drive: record record*]]

Comments:

Set **BX:CX** to the number of bytes read. The file must have been named either when you started **debug** or with the **N** (name) command. Both the **debug** invocation and the **N** command format a filename properly in the normal format of a file control block at **CS:5C**.

If you use the **L** (load) command without any parameters, **debug** loads the file into memory beginning at address **CS:100** and sets **BX:CX** to the number of bytes loaded. If you type the **L** command with an address parameter, loading begins at the memory location specified by the *address*. If you use the **L** command with all parameters included, absolute disk sectors are loaded, instead of a file.

Each *record* is taken from the specified drive: (the drive name is numeric: 0=A:, 1=B:, 2=C:, etc.). **Debug** begins loading with the first specified record, and continues until the number of sectors in the second record have been loaded.

Example:

Suppose, once you have started **debug**, that you type the following commands:

```
-NFILE.COM
```

Now, to load *file.com*, you would simply type the **L** command.

Debug would then load the file and display the **debug** prompt. Suppose now that you want to load only portions of a file or certain records from a disk. To do this, you would type the following:

```
L04BA:100 2 0F 6D
```

Debug would then load 109 (6DH) records, beginning with logical record number 15, into memory beginning at address **04BA:0100**. Then once it had loaded the records, **debug** would simply return the hyphen (-) prompt.

Using the Load command

If the file has an *.exe* extension, it would be relocated to the load address specified in the header of the *.exe* file. The *address* parameter is always ignored for *.exe* files. The header itself is stripped off the *.exe* file before it is loaded into memory. So, the size of an *.exe* file on disk will differ from its size in memory.

If the file was named by the N (name) command, or specified when you started **debug**, as a *.hex* file, then typing the L command with no parameters would cause **debug** to load the file beginning at the address specified in the *.hex* file. If the L command included the option *address*, **debug** would add the address specified in the L command to the *address* found in the *.hex* file to determine the start address at which to load the file.

(M)ove

Move**Purpose:**

Moves the block of memory specified by *range* to the location beginning at the specified *address*.

Syntax:

Mrange address

Comments:

Overlapping moves (i.e., moves where part of the block overlaps some of the current addresses) are always performed without loss of data. Addresses that could be overwritten are moved first. For moves from higher to lower addresses, the sequence of events is to first move the data beginning at the block's lowest address and then work toward the highest. For moves from lower to higher addresses, the sequence is to first move the data beginning at the block's highest address and then work toward the lowest.

Notice that if the addresses in the block being moved will not have new data written to them, the data in the block *before the move* will remain. The **M** (move) command copies the data from one area into another, in the sequence described, and writes over the new addresses. This action is why the sequence of the move is important.

Example:

**Using the Move
command**

Suppose you type the following command:

```
MCS:100 110 CS:500
```

In response, **debug** would first move address CS:110 to CS:510, then CS:10F to CS:50F, and so on until it has moved CS:100 to CS:500. To review the results of the move, you could type the **D** (dump) command, using the same *address* you used with the **M** command.

Name

(N)ame

Purpose:

Sets filenames.

Syntax:

Nfilename [*filename*...]

Comments:

The **N** (name) command performs two functions. First, it assigns a filename for a later **L** (load) or **W** (write) command. So if you start **debug** without naming a file to be debugged, you must type the command *Nfilename* before a file can be loaded. Second, the **N** command assigns filename parameters to the file being debugged. In this case, **N** accepts a list of parameters used by the file being debugged.

Notice that these two functions overlap. Consider, for example, the following set of **debug** commands:

```
-NFILE1.EXE
-L
-G
```

The **N** command would use these commands to perform the following steps:

- 1 It would first assign the filename *file1.exe* to the file to be used in any later **L** or **W** commands.
- 2 It would also assign the *file1.exe* filename to the first filename parameter used by any program that is later debugged.
- 3 The **L** command would then load *file1.exe* into memory.
- 4 The **G** (go) command would cause *file1.exe* to be run with *file1.exe* as the single filename parameter (that is, *file1.exe* would be run as if *file1.exe* had been typed at the command level).

A more useful chain of commands might look like this:

```
-NFILE1.EXE
-L
-NFILE2.DAT FILE3.DAT
-G
```

In this example, the N command sets *file1.exe* as the filename for the subsequent L command, which loads *file1.exe* into memory. The N command is then used again, this time to specify the parameters to be used by *file1.exe*. Finally, when the G command is run, *file1.exe* is executed as if *file1 file2.dat file3.dat* had been typed at the MS-DOS command level.

Notice that if you were to execute a W command now, then *file1.exe*—the file being debugged—would be saved with the name *file2.dat*. To avoid this kind of result, you should always execute an N command before either an L or W command.

There are four regions of memory that can be affected by the N command:

Memory location	Contents
CS:5C	FCB for file 1
CS:6C	FCB for file 2
CS:80	Count of characters
CS:81	All characters typed

The first filename parameter that you specify for the N command has a file control block (FCB) set up at CS:5C. If you name a second filename parameter, an FCB is set up for this parameter beginning at CS:6C. The number of characters typed in the N command (exclusive of the first character, N) is given at location CS:80.

The actual stream of characters given by the N command (again, exclusive of the letter N) begins at CS:81. Notice that this stream of characters may contain switches and delimiters that would be legal in any command typed at the MS-DOS command level.

AUsing the Name command

A typical use of the N command is as follows:

```
DEBUG PROG.COM
-NPARAM1 PARAM2/C
-G
-
```

In this case, the G command executes the file in memory as if you had typed the following command line:

```
PROG PARAM1 PARAM2/C
```

Testing and debugging therefore reflect a normal run-time environment for *prog.com*.

Output

(O)utput

Purpose:

Sends the specified *byte* to the output port specified by *value*.

Syntax:

Ovalue byte

Comment:

A 16-bit port address is allowed.

Example:

Suppose you want **debug** to output the byte value 4F to output port 2F8. To do this, you could simply type the following command:

```
O2F8 4F
```

**Using the Output
command**

(P)roceed

Proceed**Purpose:**

Executes a loop, a repeated string instruction, a software interrupt, or a subroutine call to completion.

Syntax:

P[=*address*] [*number*]

where:

address is the location of the first instruction to be executed.

number is the number of instructions to execute.

Comments:

The **P** (proceed) command transfers control from **debug** to the target program. The program executes without interruption until the loop, repeated string instruction, software interrupt, or subroutine call at *address* is completed, or until the specified number of machine instructions has been executed. Control then returns to **debug**, which displays the contents of the target program's registers and the status of the flags.

If the *address* parameter does not include an explicit segment, **debug** uses the target program's CS register; if *address* is omitted entirely, execution begins at the address specified by the target's CS:IP registers. The *address* parameter must be preceded by an equal sign (=) to distinguish it from *number*.

If the instruction at *address* is not a loop, a repeated string instruction, a software interrupt, or a subroutine call, the **P** command functions just like the **T** (trace) command. The optional *number* parameter specifies the number of instructions to be executed before control returns to **debug**. If *number* is omitted, **debug** executes only one instruction. After each instruction is executed, **debug** displays the contents of the target program's registers, the status of the flags, and the next instruction to be executed.

Warning

The **P** command cannot be used to trace through ROM.

Example:

Assume that the target program's location CS:143FH contains a **call** instruction. To execute the subroutine that is the destination of **call** and then return control to **debug**, type:

-P =143F

**Using the Proceed
command**

Quit

(Q)uit

Purpose:

Terminates the **debug** utility.

Syntax:

Q

Comments:

The **Q** (quit) command takes no parameters and exits **debug** without saving the file you're currently working with. You are returned to the MS-DOS command level.

Example:

To end the debugging session, simply type the following and press the **ENTER** key:

q

Debug terminates, and control returns to the MS-DOS command level.

Using the Quit command

(R)register

Register**Purpose:**

Displays the contents of one or more CPU registers.

Syntax:

R[*register-name*]

Comments:

If you do not type a *register-name*, the **R** (register) command dumps the register storage area and displays the contents of all registers and flags.

If you do type a *register-name*, the 16-bit value of that register is displayed in hexadecimal, and a colon then appears as a prompt. You can now either type a *value* to change the register, or press the **ENTER** key if you don't want a change.

Following is a list of the valid register-names:

AX	BP	SS	
BX	SI	CS	
CX	DI	IP	(IP and PC both refer
DX	DS	PC	to the Instruction Pointer.)
SP	ES	F	

Any other entry for *register-name* results in a **BR** error message.

If you type **F** as the *register-name*, **debug** displays each flag with a two-character alphabetic code. To change any flag, type the opposite two-letter code. The flags are then either set or cleared.

The flags are listed below with their codes for SET and CLEAR:

Flag name	Set	Clear
Overflow	OV	NV
Direction	DN (Decrement)	UP (Increment)
Interrupt	EI (Enabled)	DI (Disabled)
Sign	NG (Negative)	PL (Plus)
Zero	ZR	NZ
Auxiliary Carry	AC	NA
Parity	PE (Even)	PO (Odd)
Carry	CY	NC

Whenever you type the **RF** command, the flags are displayed (in the order shown in the previous table) in a row at the beginning of a line. At the end of the list of flags, **debug** displays a hyphen (-).

You may enter new flag values in any order as alphabetic pairs. You do not have to leave spaces between these values. To exit the **R** command, press the **ENTER** key. Any flags for which you did not specify new values remain unchanged.

If you type more than one value for a flag, **debug** returns a **DF** error message. If you enter a flag code other than one of those shown in the table above, **debug** returns a **BF** error message. In both cases, the flags up to the error in the list are changed; those flags at and after the error are not.

When you start **debug**, the segment registers are set to the bottom of free memory, the Instruction Pointer is set to 0100H, all flags are cleared, and the remaining registers are set to zero.

**Using the Register
command**
Examples:

If you type the following command, **debug** displays all registers, flags, and the decoded instruction for the current location:

R

If the location is CS:11A, for example, the display will look similar to this:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A  NV UP DI NG NZ AC PE NC
04BA:011A CD21                INT 21
```

If you then type the following command, **debug** will display these flags:

```
RF
NV UP DI NG NZ AC PE NC - _
```

Now, you could type any valid flag designation, in any order, with or without spaces. For example,

```
NV UP DI NG NZ AC PE NC - PLEICY
```

In response, **debug** would display the **debug** prompt. To see the changes, type either the **R** or **RF** command:

```
RF
NV UP EI PL NZ AC PE CY - _
```

Press the **ENTER** key to leave the flags this way or to specify different flag values.

Search

(S)earch

Purpose:

Searches the specified range for the specified lists of bytes.

Syntax:

String list

Comments:

The *list* may contain one or more bytes, each separated by a space or comma. If the *list* contains more than one byte, only the first address of the byte string is returned. If the *list* contains only one byte, all addresses of the byte in the *range* are displayed.

Example:

Suppose you type the following command:

```
SCS:100 110 41
```

Debug would display a response similar to

```
04BA:0104
04BA:010D
-type:
```

Using the Search command

(T)race**Trace****Purpose:**

Executes one instruction and displays the contents of all registers, flags, and the decoded instruction.

Syntax:

T[=*address*] [*value*]

Comments:

If you include the =*address* option in the T (trace) command, tracing occurs at the specified =*address*. The *value* option causes **debug** to execute and trace the number of steps specified by *value*.

The T command uses the hardware trace mode of the 8086 or 8088 microprocessor. Consequently, you may also trace instructions stored in ROM (Read Only Memory).

Example:

Using the Trace command

Suppose you type the following command:

T

In response, **debug** would return a display of the registers, flags, and decoded instruction for that one instruction. Assuming, for this example, that the current position is 04BA:011A, **debug** might return the following display:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A NV UP DI NG NZ AC PE NC
04BA:011A CD21 INT 21
```

If you type the following command, **debug** executes sixteen (10 hex) instructions beginning at 011A in the current segment, and then displays all registers and flags for each instruction as it is executed. The display scrolls away until the last instruction is executed, and then stops. Now you can see the register and flag values for the last few instructions performed:

T=011A 10

Remember that if you want to study the registers and flags for any instruction (at any time), you can press **CONTROL+S** to stop the display from scrolling.

Unassemble

 (U)nassemble

Purpose:

Disassembles bytes and displays the source statements that correspond to them, with addresses and byte values.

Syntax:

U[*range*]

Comments:

The display of disassembled code looks like a listing for an assembled file. If you type the U (unassemble) command without parameters, 20 hexadecimal bytes are disassembled at the first address after that displayed by the previous U command. If you type the U command and include the *range* parameter, **debug** disassembles all bytes in *range*. But if you specify *range* only as an address, 20H bytes are disassembled.

Example:

Suppose you type the following command:

```
U04BA:100 L10
```

In response, **debug** would disassemble 16 bytes, beginning at address 04BA:0100:

```
04BA:0100 206472 AND [SI+72],AH
04BA:0103 69 DB 69
04BA:0104 7665 JBE 016B
04BA:0106 207370 AND [BP+DI+70],DH
04BA:0109 65 DB 65
04BA:010A 63 DB 63
04BA:010B 69 DB 69
04BA:010C 66 DB 66
04BA:010D 69 DB 69
04BA:010E 63 DB 63
04BA:010F 61 DB 61
```

**Using the Unassemble
command**

Now, suppose you type the following:

```
U04ba:0100 0108
```

The display would now show

```
04BA:0100 206472 AND [SI+72],AH
04BA:0103 69 DB 69
04BA:0104 7665 JBE 016B
04BA:0106 207370 AND [BP+DI+70],DH
```

If the bytes in some addresses are altered, the disassembler alters the instruction statements. You can then type the U command for the changed locations, for the new instructions viewed, and for the disassembled code used to edit the source file.

Write

(W)rite

Purpose:

Writes the file being debugged to a disk file.

Syntax:

W[*address* [*drive: record record*]]

Comments:

If you do not use parameters with the **W** (write) command, **BX:CX** must already be set to the number of bytes to be written; the file is written beginning from **CS:100**. If you type the **W** command with just an *address*, the file is written beginning at that *address*. If you have used a **G** (go) or **T** (trace) command, you must reset **BX:CX** before using the **W** command without parameters.

Notice that if a file is loaded and modified, the name, length, and starting address are all set correctly to save the modified file (as long as the length has not changed).

You must have named the file either with the initial **debug** startup command or with the **N** (name) command (refer to the **N** command described earlier in this chapter). Both the **debug** startup command and the **N** command properly format a filename in the normal format of a file control block at **CS:5C**.

If you include parameters when you use the **W** command, the write begins from the memory address specified; the file is written to the specified *drive*: (the drive name is numeric: 0=A:, 1=B:, 2=C:, etc.). **Debug** writes the file beginning at the logical record number specified by the first *record*. **Debug** then continues to write the file until the number of sectors specified in the second *record* have been written.

Warning

Writing to absolute sectors is *extremely risky* because the process bypasses the file handler.

Examples:

If you type the following command, **debug** writes out the contents of memory to the disk in drive B, beginning with the address **CS:100**. The data written out starts in the disk logical record number 37H and consists of 2BH records.

```
WCS:100 1 37 2B
```

When the write is complete, **debug** displays its prompt again.

Using the Write command

**(XA) Allocate
Expanded Memory**

Allocate Expanded Memory**Purpose:**

Allocates a specified number of pages as expanded memory (EMS) pages. Displays "Handle created" if successful.

Syntax:

XA *count*

where:

count is the hexadecimal number of 16K pages to allocate.

Comments:

If the specified number of pages is available, **debug** displays a message indicating the number of the handle created. If it is not, **debug** displays an error message.

Example:

To allocate eight EMS pages, type the following:

```
XA 8
```

If eight pages of EMS memory are available, **debug** displays a message similar to the following:

```
Handle created=0003
```

Deallocate Expanded Memory

(XD) Deallocate Expanded Memory

Purpose:

Deallocates a handle.

Syntax:

XD handle

where:

handle is the handle to be deallocated.

Comments:

If the handle number is valid, **debug** deallocates the handle and displays a message.

Example:

To deallocate handle 0003, type the following:

```
xd 0003
```

If the handle is deallocated successfully, **debug** displays the following message:

```
Handle 0003 deallocated
```

**(XM) Map Expanded
Memory Pages**

Map Expanded Memory Pages**Purpose:**

Maps an EMS logical page, belonging to the specified EMS handle, into an EMS physical page.

Syntax:

XM lpage ppage handle

where:

lpage is the number of the page among the pages allocated to the *handle* specified.

ppage is the number of the physical page into which *lpage* is to be mapped.

Example:

To map logical page 5 of EMS handle 0003 to physical page 2, type the following:

```
xm 5 2 0003
```

If the operation is successful, **debug** displays the following message:

```
Logical page 05 mapped to physical page 02
```

Get Expanded Memory Status

(XS) Get Expanded Memory Status

Purpose:

Displays EMS information.

Syntax:

XS

Comments:

The display has the following format:

```
Handle %1 has %2 pages allocated
Physical page %1 = Frame segment %2
%1 of a total %2 EMS pages have been allocated
%1 of a total %2 EMS handles have been allocated
```

Example:

To display EMS information, type the following:

XS

Debug will then display information similar to the following:

```
Handle 0000 has 0000 pages allocated
Handle 0001 has 0002 pages allocated

Physical page 00 = Frame segment C000
Physical page 01 = Frame segment C400
Physical page 02 = Frame segment C800
Physical page 03 = Frame segment CC00
  2 of a total 80 EMS pages have been allocated
  2 of a total FF EMS handles have been allocated
```

Debug Error Messages

Debug Error Messages

During a **debug** session, you may receive any of the following error messages. Each error ends the **debug** command under which it occurred, but does not end **debug** itself.

Error**Code Definition****BF Bad Flag**

You attempted to change a flag, but the characters you typed were not one of the acceptable pairs of flag values. See the **R** (register) command for the list of acceptable flag entries.

BP Too many breakpoints.

You specified more than ten breakpoints as parameters to the **G** (go) command. Retype the **G** command using ten or fewer breakpoints.

BR Bad register.

While using the **R** command, you typed an invalid register name. See the **R** command for the list of valid register names.

DF Double flag.

You typed two values for one flag. You may specify a flag value only once per **RF** command.

Appendix A

Instructions for Users with Single Floppy Disk Systems

If you have only one floppy disk drive, you can still type MS-DOS commands just as if you had two disk drives on your system.

Just think of your one-drive system as having *two* drives (drive A and drive B). But instead of A and B representing two physical drives, they represent disks.

Just remember that when you specify drive B, when the drive A disk was used last, MS-DOS prompts you to insert the disk for drive B. For example,

```
A> copy command.com b:
Insert diskette for drive B:
and strike any key when ready
1 File(s) copied
A>_
```

If you specify drive A when the drive B disk was used last, MS-DOS asks you to change disks again, this time prompting you to insert the drive A disk.

When using a batch file to execute commands, you follow the same procedure. MS-DOS waits for you to insert the appropriate disk and press any key before it continues.

Important

The letter displayed in the system prompt represents the default drive where MS-DOS looks to find a file whose name is entered without a drive name; this letter does *not* represent the last disk used.

Assume, for example, that A is the default drive. If the last command performed was `dir b:`, MS-DOS would act as if the drive B disk is still in the drive. The system prompt, however, is still `A>`, because A is still the default drive. If you type the `dir` command, MS-DOS prompts you for the drive A disk, because it is the default drive and you did not specify another drive in the `dir` command.

Appendix B

How to Configure Your System

What is a Configuration File?

The configuration file *config.sys* is a file that contains certain commands that MS-DOS checks when you start up your computer. Each time you start MS-DOS, it searches the root directory of the drive in which it was started for a file named *config.sys*.

The *config.sys* file lets you change your system's default configuration settings. For example, you can add installable device drivers to your system by including special commands in your *config.sys* file.

You can use the *dir* command to see if the *config.sys* file is already on drive C. If it is on the disk, you can use the *type* command to display it. For more information about the *dir* and *type* commands, see "Chapter 3, MS-DOS Commands".

If *config.sys* already exists, you may want to include other commands in the file. For example, you may want to configure drive C for a new device, such as a mouse or an external drive.

If your MS-DOS disk does not have a *config.sys* file, you can use the MS-DOS line editor, *Edlin*, to create one and then save it on the MS-DOS disk in your root directory. If *config.sys* already exists and you want to change it, you can use *Edlin* to edit it.

Checking for *config.sys*

Config.sys Commands

The following table briefly describes the purpose of each configuration command:

The configuration commands

Command	Purpose
break	Sets CONTROL+C check.
buffers	Sets the number of sector buffers.
country	Allows for international time, date, currency, capitalization, collating sequence, folding format, and decimal separator.
device	Installs the device driver in the system.
drivparm	Defines parameters for block devices.
fcbs	Specifies the number of FCBs that can be open concurrently.
files	Sets the number of open files that can access certain MS-DOS system calls.
install	Executes special MS-DOS commands during <i>config.sys</i> processing.
lastdrive	Sets the maximum number of drives you may access.
rem	Adds comments to <i>config.sys</i> files.
shell	Begins execution of the shell from a specific file (usually <i>command.com</i>).
stacks	Supports the dynamic use of data stacks.

Sample Config.sys File

A typical configuration file might look like this:

```

buffers=30
device=c:\dev\network.sys
break=on
install=c:\dos\fastopen c:100
shell=c:\bin\command.com c:\bin /p
lastdrive=z

```


The following explains how this sample *config.sys* file configures the system:

Command	Purpose
buffers	Sets the number of buffers in memory to 30.
device	Sets a searchpath of <i>c:\dev\network.sys</i> to find the device driver being added to the system. In this case, it is network software. Generally, you will receive a disk with some installation software when you purchase a new device. Make sure that the device driver is in the directory that you specify in the <i>device</i> command.
break	Makes it possible to use CONTROL+C in real mode to stop a program, depending on the program that you are running.
install	Invokes a fastopen command from <i>config.sys</i> .
shell	Sets the MS-DOS command processor to the <i>command.com</i> file located in the <i>bin</i> directory on the disk in drive C. <i>c:\bin</i> tells the command processor where to look for <i>command.com</i> if it needs to reread the disk. The /p switch is used by the <i>command.com</i> shell. For more information about <i>command.com</i> and its switches, see "Chapter 3, MS-DOS Commands".
lastdrive	Sets the last available label for a logical or physical drive to z . In other words, on this computer, letters from a to z are available as labels for logical or physical disk drives.

The following pages describe each configuration command in detail.

Break

Break

Purpose:

Sets **CONTROL+C** check.

Syntax:

break

or

break=on

or

break=off

Default:

break=off

Comments:

You can type **break** to display the break status.

Depending on the program you are running, you may use **CONTROL+C** to stop an activity (for example, to stop sorting a file). Normally, MS-DOS checks to see whether you have pressed **CONTROL+C** only while it is reading from the keyboard or writing to the screen or printer. Therefore, setting **break** to **on** extends **CONTROL+C** checking to other functions, such as disk reading and writing.

Example:

To turn on **CONTROL+C** checking, put the following line in your *config.sys* file:

```
break=on
```

Initiating **CONTROL+C
checking**

Buffers

Buffers

Purpose:

Allows you to set the number of disk buffers that MS-DOS allocates in memory at the time you start the system.

Syntax:

buffers=*n*[/*m*]/*x*]

where:

n is the number of disk buffers, from 1 to 99. If */x* is specified, the maximum number of disk buffers is 10,000 or the largest number of buffers that fit in memory, whichever is less.

m is the maximum number of sectors that can be read or written in one I/O operation (1-8). The default is 1.

Default:

Configuration	Buffers
For a base system	2
Any disk over 360 kilobytes	3
128 to 255 kilobytes RAM	5
256 to 511 kilobytes RAM	10
512 kilobytes or more RAM	15

Comments:

The */x* switch places the buffers in expanded memory. This switch fails if all expanded memory is used.

A *disk buffer* is a block of memory that MS-DOS uses to hold data when reading or writing.

For applications such as word processors, a number between 10 and 20 provides the best performance. If you plan to create a lot of subdirectories, you may even want to increase the buffers value to between 20 and 30. Remember, though, that buffers take up approximately 528 bytes of space, so the more buffers you have, the less memory you will have available for applications.

Note

Feel free to experiment with different **buffer** settings to see how different values affect the way your personal computer operates.

Example:

Setting the number of disk buffers

To create 20 disk buffers, put the following line in your *config.sys* file:

```
buffers=20
```

Country

Country

Purpose:

Country allows MS-DOS to use international time, date, currency, case conversions, folding format, and decimal separator.

Syntax:

country=xxx[,yyy][,drive:]filename]

where:

xxx is the country code.

yyy is the code page for the country.

filename is a file containing country information.

Default:

Unless otherwise specified, United States settings are assumed.

If you do not specify *filename*, MS-DOS uses the *country.sys* file for country-specific information.

Comments:

This configuration command identifies to MS-DOS which country's character set you intend to use. The following table lists each country (or language) supported by MS-DOS 4.0. The table also lists the code pages that may be used with a specific country code. For example, if you enter country code 003, you can only enter code pages 437 and 850 on the same statement. You could not enter country code 003 with code page 863.

Country or language	Country code	Code pages
United States	001	437,850
Canadian-French	002	863,850
Latin America	003	437,850
Netherlands	031	437,850
Belgium	032	437,850
France	033	437,850
Spain	034	437,850
Italy	039	437,850
Switzerland	041	437,850
United Kingdom	044	437,850
Denmark	045	865,850
Sweden	046	437,850
Norway	047	865,850

Country or language	Country code	Code pages
Germany	049	437,850
English (International)	061	437,850
Portugal	351	860,850
Finland	358	437,850
Arabic	785	437
Israel	972	437
Japan	081	932,850,437
Korea	082	934,850,437
Peoples Republic of China	086	936,850,437
Taiwan	088	938,850,437

The country code affects four MS-DOS commands, **date**, **backup**, **restore**, and **time**. The following table lists the date and time formats related to each country (or language group).

For each country, the Date format column shows how MS-DOS would display January 3, 1989, and the Time format column shows how MS-DOS would display 5:35 p.m. (with zero seconds and zero hundredths of seconds).

Country or language	Country code	Date format	Time format
United States	001	01-03-1989	5:35:00.00
Canadian-French	002	03-01-1989	17:35:00,00
Latin America	003	03/01/1989	17:35:00.00
Netherlands	031	03-01-1989	17:35:00,00
Belgium	032	03/01/1989	17:35:00,00
France	033	03/01/1989	17:35:00,00
Spain	034	03/01/1989	17:35:00,00
Italy	039	03/01/1989	17:35:00,00
Switzerland	041	03.01.1989	17.35.00.00
United Kingdom	044	03-01-1989	5:35:00.00
Denmark	045	03-01-1989	17.35.00.00
Sweden	046	1989-01-03	17.35.00.00
Norway	047	03.01.1989	17.35.00.00
Germany	049	03.01.1989	17.35.00.00
English (International)	061	03-01-1989	5:35:00.00

Country or language	Country code	Date format	Time format
Portugal	351	03/01/1989	17:35:00,00
Finland	358	03.01.1989	17.35.00,00
Israel	972	03 01 1989	17:35:00.00
Arabic	785	03/01/1989	5:35:00,00
Japan	081	1989-01-03	17:35:00.00
Korea	082	1989-01-03	17:35:00.00
Peoples Republic of China	086	1989-01-03	17:35:00.00
Taiwan	088	01-03-1989	17:35:00.00

Appendix E, “How to Use Code Pages”, provides additional information on using the `country` command and code pages.

Example:

The following example sets `country` to France (=033) and converts international currency, time, date, and case to French conventions:

```
country=033
```

Setting Country conversions

Device

Device**Purpose:**

Installs the specified device driver on the system list.

Syntax:

device=[drive:][path]filename[argument]

where:

argument includes any switches accepted by filename.

Default:

None

Comments:

The standard installable device drivers provided with MS-DOS are *ansi.sys*, *display.sys*, *driver.sys*, *printer.sys*, *ramdrive.sys*, and *smartdrv.sys*. For more information on these installable device drivers, see Appendix C, "Installable Device Drivers".

Other installable device drivers

If you purchase a new device, like a mouse or a scanner, you generally will receive device driver software with that device. These installable device drivers can be installed using the **device** command. Once you have installed a device driver, make sure that the device driver is in the directory that you specify in any device commands.

Note .

The device drivers *country.sys* and *keyboard.sys* are loaded automatically by MS-DOS. Do not try to load either of these with the **device** command. If you do, it will hang your system (that is, MS-DOS will not start).

Example:

If you plan to use the ANSI escape sequences described in Appendix C, "Installable Device Drivers", you should create a *config.sys* file that contains the following command:

```
device=ansi.sys
```

This command causes MS-DOS to replace all keyboard input and screen output support with the ANSI escape sequences.

Drivparm

Drivparm

Purpose:

This command allows you to define parameters for block devices when you start MS-DOS, overriding the original MS-DOS device driver settings. If you want to create a new logical drive and associate it with a physical drive, see the *driver.sys* section in Appendix C, “Installable Device Drivers”. By contrast, **driveparm** will modify the parameters of an existing physical drive and not create a new logical drive.

Syntax:

driveparm=*/d:number* [*/c*]/*/f:factor*] [*/h:heads*] [*/i*] [*/n*]/*/s:sectors*]
[*/t:tracks*]

Default:

None

Comments:

Setting **driveparm** overrides any previous block device driver definitions.

The following list describes how each switch is used:

Switch	Function
<i>/d:number</i>	Physical drive number, ranging from 0 to 255. This means that drive number 0=A, 1=B, 2=C, etc.
<i>/c</i>	Indicates that the physical disk drive referenced has change-line support in the hardware. This means that the physical disk drive can sense that the drive door has been opened. If the door is open, the device driver will assume that the drive does not have a disk in it yet. If you specify this switch, MS-DOS assumes that the physical disk drive has the ability to support change-line error detection. See the documentation for your disk drive to see if it supports change-line.

Switch	Function
<i>/f:factor</i>	Specifies the device type (form factor). The default value is 2. factor=form factor index, where 0 = 160/180K bytes, or 0 = 320/360K bytes 1 = 1.2 megabytes 2 = 720K bytes (3.5-inch disk) 5 = Hard disk 6 = Tape drive 7 = 1.44 megabytes (3.5-inch disk)

The default values for the following switches depend upon the form factor specified with the */f:* switch. If you do not specify the */f:* switch, **drivparm** uses a default of 720K bytes (3.5-inch disk).

Switch	Function
<i>/h:heads</i>	Maximum head number, ranging from 1 to 99. The default value is 2.
<i>/i</i>	Specifies an electrically-compatible 3.5-inch disk drive. (Electrically-compatible drives are installed in your computer and use your existing floppy controller.) The drive is connected to the floppy controller as either A or B. Use <i>/i</i> if your computer's ROM-BIOS does not support 3.5-inch disk drives. (Some AT-compatible systems do not have a ROM-BIOS that supports 3.5-inch disk drives.)
<i>/n</i>	Specifies a nonremovable block device.
<i>/s:sectors</i>	Number of sectors per track, ranging from 1 to 99. The default value is 9.
<i>/t:tracks</i>	Number of tracks per side on the block device, ranging from 1 to 999.

Example:

Suppose your computer has an internal tape drive on drive D that is configured at startup to write 20 tracks of 40 sectors per track. If you want to reconfigure this tape drive to write 10 tracks of 99 sectors each, you can put the following line in your *config.sys* file:

```
drivparm=/d:3 /f:6 /h:1 /s:99 /t:10
```

This command line overrides the default device driver settings, and supports a tape drive as drive D (in this case the logical and physical drive numbers are identical). This tape drive has 1 head and supports a tape format of 10 tracks and 99 sectors per track. (This assumes that the device driver for the tape device supports this configuration of tracks and sectors.) So to create a tape that you can read on another computer, one which can read only this alternate format, you might want to use this method.

Reconfiguring a tape drive

FCBS

FCBS**Purpose:**

Allows you to determine the number of File Control Blocks (FCBs) that can be open concurrently.

Syntax:

fcbs=x,y

where:

x is the number of files that File Control Blocks can open at one time.

y is the number of files opened by FCBs that MS-DOS cannot close automatically.

Default:

fcbs=4,0

Comments:

A *file control block* is a data structure in real mode used to control open files.

The allowed values for *x* range from 1 to 255. The allowed values for *y* also range from 1 to 255. If an application tries to open more than *x* files by FCBs, then all but the first *y* files may be closed by MS-DOS.

Note

The preferred method of accessing files is to use file handles instead of file control blocks. However, some older applications that you may want to use may require you to use the **fcbs** command in your *config.sys* file. Thus, you should only use the **fcbs** command if an application requires you to do so.

Example:

To open up to four files by FCBs and to protect the first two files from being closed, put the following line in your *config.sys* file:

fcbs=4,2

Setting the number of open file control blocks

Files

Files

Purpose:

Sets the number of open files that the MS-DOS system calls can access.

Syntax:

files=x

where:

x is the number of open files that the system calls can access.

Default:

files=8

Comments:

The valid values for *x* range from 8 to 255. The maximum number of files that one program can have open at a time is 65,534.

A process must issue MS-DOS system call 67H to set the number of file handles it can have open at any given time. Notice, however, that this just increases the number of handles, not the number of files, that can be opened. The number of files that can be opened is still limited by the **files=x** parameter.

Example:

To let MS-DOS open 20 files at one time, put the following line in your *config.sys* file:

```
files=20
```

**Setting the number of
open files**

Install

Install**Purpose:**

Executes a *fastopen*, *keyb*, *nlsfunc*, or *share* command during *config.sys* processing.

Syntax:

install=[drive:][path]filename [parms]

where:

parms is the optional command line to pass to the filename. The filename must be *fastopen.exe*, *key.exe*, *nlsfunc.exe*, or *share.exe*.

Default:

None

Example:

You can invoke the **fastopen** command only once without restarting MS-DOS. To prevent the command from being reissued each time you run *autoexec.bat*, add your **fastopen** command to *config.sys* with an **install** command. For example:

```
install=fastopen.exe c:100
```

Lastdrive

Lastdrive

Purpose:

Sets the maximum number of drives you may access.

Syntax:

lastdrive=x

where:

x can be any letter from A to Z.

Default:

lastdrive=e

Comments:

The *x* value represents the last valid drive that MS-DOS will accept. The minimum number is equal to the number of drives you have installed on your computer.

This command is useful only in a network environment. At start-up, MS-DOS recognizes five drives you have on your system. To make any extra drives defined by **lastdrive** valid, a network redirection must occur.

Notice that MS-DOS allocates a data structure for each drive that you specify, so you shouldn't specify more drives than are necessary.

Example:

The following command sets the last drive to M, unless you have added an external logical device with *driver.sys*. For information about *driver.sys*, see Appendix C, "Installable Device Drivers".

```
lastdrive=m
```

Rem

Rem**Purpose:**

Allows comments to be entered in *config.sys*.

Syntax

rem *comment*

where:

comment is any string.

Default:

None

Example:

The following example identifies the country code used in the **country** command:

```
rem Set country code to France
country=033
```

Shell

Shell**Purpose:**

Begins execution of the shell (top-level command processor) from a file defined by the specified pathname.

Syntax:

shell=[*drive:*][*path*]*filename* [*parms*]

where:

filename is a command processor program.

parms is the optional command line to pass the filename.

Default:

The default command processor for MS-DOS is *command.com*. Refer to the description of **command** in Chapter 3, "MS-DOS Commands", for switch information.

Comments:

Instead of reading the standard *command.com*, MS-DOS starts the processor specified in *filename*.

System programmers who write their own command processors (instead of using the MS-DOS file, *command.com*) should use the **shell** command to specify the name of their shell program.

MS-DOS sets the COMSPEC environment variable equal to the *drive*:, *path* and *filename* specified on the **shell** command line. This setting overrides the default value for COMSPEC (the drive and pathname of the command processor initially used to start MS-DOS). The operating system uses the COMSPEC environment setting to determine which file to use when reloading any transient part of the command processor.

Note

The **shell** command does not accept switches. However, if the new command processor does accept switches, you can include those switches in this syntax. For example, suppose **shell=newcmdp.com**. And suppose *newcmdp.com* accepts the /c, /p, and /e switches. You can include any of these switches in the **shell** command line. Thus, the following would be a valid command:

```
shell=newcmdp.com /p
```

Example:

The following command uses the file `\bin\newshell` as the command processor:

```
shell=\bin\newshell
```

Setting the command processor shell

Stacks

Stacks**Purpose:**

Supports the dynamic use of data stacks.

Syntax:

stacks=*n,s*

where:

n is the number of stacks.

s is the size of each stack.

Default:

Computer	Stacks
For IBM-PC, IBM-XT, IBM PC-Portable	0,0
For other computers	9,128

Comments:

The valid values for *n* range from 0 to 64. The valid values for *s* range from 0 to 512.

When there is a hardware interrupt, MS-DOS allocates one stack from *n* stacks specified. When **stacks=0,0**, MS-DOS will not switch stacks at interrupt time.

Examples:

If you want to allocate eight stacks of 512 bytes each for hardware interrupt handling, you would include the following command in your *config.sys* file:

```
stacks=8,512
```

Appendix C

Installable Device Drivers

Introduction

Device drivers are programs that let the operating system recognize devices that are not part of the computer. Examples of devices are a modem, a printer, a mouse, and an external disk drive. Some device drivers are already installed with MS-DOS. Other device drivers, called installable device drivers, come with MS-DOS for you to install if you need them.

This appendix describes the installable device drivers provided with MS-DOS:

- *Ansi.sys* loads the ANSI character set.
- *Display.sys* supports code-page switching on the console device.
- *Driver.sys* supports external floppy disk drives.
- *Printer.sys* provides code page support for PRN, LPT1, LPT2, and LPT3.
- *Ramdrive.sys* supports one or more RAM (virtual) drives.
- *Smartdrv.sys* adds a disk-caching program for computers that have a hard disk and expanded or extended memory.

For more information about the device configuration command, which is used to install these device drivers, see Appendix B, “How to Configure Your System”.

Ansi.sys

The *ansi.sys* installable device driver lets you use ANSI escape sequences in real mode. An *ANSI escape sequence* is a series of characters (beginning with an escape character or keystroke) developed by the American National Standards Institute (ANSI). These sequences are used to define functions for MS-DOS. Specifically, you can change graphics functions and affect the movement of the cursor.

Installing Ansi.sys

To install *ansi.sys*, include a command line of the following form in your *config.sys* file:

```
device=[drive][path]ansi.sys[/x]
```

The *ansi.sys* device driver accepts a /x switch that allows expanded keys to be remapped independently. On the 101-key keyboard, for example, there are two HOME keys. One HOME key is on the numeric keypad and the other is in the block of cursor-control keys. The two HOME keys are considered to be the same unless the /x switch is specified.

ANSI Escape Sequences Used with MS-DOS

This section lists and explains valid ANSI escape sequences for MS-DOS.

The variables listed in the escape sequences themselves are as follows:

The Ansi.sys variables

Code	Description
<i>Pn</i>	<i>Numeric parameter</i> - a decimal number that you specify with ASCII digits.
<i>Ps</i>	<i>Selective parameter</i> - a decimal number that you use to select a subfunction. You may specify more than one subfunction by separating the parameters with semicolons.
<i>Pl</i>	<i>Line parameter</i> - a decimal number that you specify with ASCII digits.
<i>Pc</i>	<i>Column parameter</i> - a decimal number that you specify with ASCII digits.

Sequence	Function	
ESC [<i>Pl</i> ; <i>Pc</i> H	Cursor Position (CUP)	Changing the cursor position
ESC [<i>Pl</i> ; <i>Pc</i> F	Horizontal & Vertical Position (HVP) CUP and HVP move the cursor to the position specified by the parameters. When no parameters are provided, the cursor moves to the home position (the upper-left corner of the screen).	
ESC [<i>Pn</i> A	Cursor Up (CUU) This sequence moves the cursor up <i>Pn</i> lines without changing columns. If the cursor is already on the top line, MS-DOS ignores the CUU sequence.	
ESC [<i>Pn</i> B	Cursor Down (CUD) This sequence moves the cursor down <i>Pn</i> lines without changing columns. If the cursor is already on the bottom line, MS-DOS ignores the CUD sequence.	
ESC [<i>Pn</i> C	Cursor Forward (CUF) The CUF sequence moves the cursor forward <i>Pn</i> columns without changing lines. If the cursor is already in the far right column, MS-DOS ignores the CUF sequence.	
ESC [<i>Pn</i> D	Cursor Backward (CUB) This escape sequence moves the cursor back <i>Pn</i> columns without changing lines. If the cursor is already in the far left column, MS-DOS ignores the CUB sequence.	
ESC [6 n	Device Status Report (DSR) The console driver outputs an RCP sequence when it receives the DSR escape sequence.	
ESC [s	Save Cursor Position (SCP) The console driver saves the current cursor position. This position can be restored with the RCP sequence.	

	Sequence	Function
	ESC [u	Restore Cursor Position (RCP) This sequence restores the cursor position to the value it had when the console driver received the SCP sequence.
Affecting erase functions	ESC [2 J	Erase Display (ED) The ED sequence erases the screen. The cursor then goes to the home position.
	ESC [K	Erase Line (EL) This sequence erases from the cursor to the end of the line (including the cursor position).
Affecting screen graphics	ESC [<i>Ps</i> ; ... ; <i>Ps</i> m	Set Graphics Rendition (SGR) The SGR escape sequence calls the graphic functions specified by the following numeric parameters. These functions remain until the next occurrence of an SGR escape sequence.

Graphics Functions

- 0 All attributes off
- 1 Bold on
- 4 Underscore (monochrome display only)
- 5 Blink on
- 7 Reverse video on
- 8 Concealed on

Foreground colors:

- 30 Black
- 31 Red
- 32 Green
- 33 Yellow
- 34 Blue
- 35 Magenta
- 36 Cyan
- 37 White

Sequence	Function
	Background colors:
	40 Black
	41 Red
	42 Green
	43 Yellow
	44 Blue
	45 Magenta
	46 Cyan
	47 White
	Parameters 30 through 47 meet the ISO 6429 standard.
ESC [= P s h	Set Mode (SM)
ESC [= h	The SM escape sequence changes the
ESC [= 0 h	screen width or type to one of the
ESC [? 7 h	following numeric parameters:
	Screen width parameters
	0 40 25 B&W
	1 40 25 color
	2 80 25 B&W
	3 80 25 color
	4 320 200 color
	5 320 200 B&W
	6 640 200 B&W
	7 Wraps at the end of each line
	14 640 200 color
	15 640 350 mono
	16 640 350 color
	17 640 480 color
	18 640 480 color
	19 320 200 color
ESC [= P s l	Reset Mode (RM)
ESC [= l	Parameters for RM are the same as for
ESC [= 0 l	SM (Set Mode), except parameter 7 resets
ESC [? 7 l	the mode that causes wrapping at the end of each line.

Sequence Function

ESC[*code*;*string*;*...p*

Allows redefinition of keyboard keys to a specified string where:

string is either the ASCII code for a single character or a string contained in quotation marks. For example, both 65 and "A" can be used to represent an uppercase A.

code is one or more of the following values that represent keyboard keys. Semicolons shown in this table must be entered in addition to the required semicolons in the command line.

Key	Code			
	Alone	Shift	Ctrl	Alt
F1	0;59	0;84	0;94	0;104
F2	0;60	0;85	0;95	0;105
F3	0;61	0;86	0;96	0;106
F4	0;62	0;87	0;97	0;107
F5	0;63	0;88	0;98	0;108
F6	0;64	0;89	0;99	0;109
F7	0;65	0;90	0;100	0;110
F8	0;66	0;91	0;101	0;111
F9	0;67	0;92	0;102	0;112
F10	0;68	0;93	0;103	0;113
F11	0;133	0;135	0;137	0;139
F12	0;134	0;136	0;138	0;140
Home	0;71	55	0;119	—
Up Arrow	0;72	56	—	—
Pg Up	0;73	57	0;132	—
Left Arrow	0;75	52	0;115	—
Down Arrow	0;77	54	0;116	—
End	0;79	49	0;117	—
Pg Dn	0;81	51	0;118	—
Ins	0;82	48	—	—
Del	0;83	46	—	—
PrtSc	—	—	0;114	—

Key	Code			
	Alone	Shift-	Ctrl-	Alt-
A	97	65	1	0;30
B	98	66	2	0;48
C	99	67	3	0;46
D	100	68	4	0;32
E	101	69	5	0;18
F	102	70	6	0;33
G	103	71	7	0;34
H	104	72	8	0;35
I	105	73	9	0;23
J	106	74	10	0;36
K	107	75	11	0;37
L	108	76	12	0;38
M	109	77	13	0;50
N	110	78	14	0;49
O	111	79	15	0;24
P	112	80	16	0;25
Q	113	81	17	0;16
R	114	82	18	0;19
S	115	83	19	0;31
T	116	84	20	0;20
U	117	85	21	0;22
V	118	86	22	0;47
W	119	87	23	0;17
X	120	88	24	0;45
Y	121	89	25	0;21
Z	122	90	26	0;44
1	49	33	—	0;120
2	50	64	—	0;121
3	51	35	—	0;122
4	52	36	—	0;123
5	53	37	—	0;124
6	54	94	—	0;125
7	55	38	—	0;126
8	56	42	—	0;127
9	57	40	—	0;128
0	48	41	—	0;129
-	45	95	—	0;130
=	61	43	—	0;131
Tab	9	0;15	—	—
Null	0;3	—	—	—

Examples:

To exchange the backslash and question-mark keys using literal strings to denote the keys, use the escape sequence:

```
ESC["\";"?"pESC["?";"\"p
```

To exchange the backslash and question-mark keys using each key's ASCII value to denote the key, use the escape sequence:

```
ESC[92;63pESC[63;92p
```

To restore the backslash and question-mark keys to their original meanings, use the escape sequence:

```
ESC[92;63pESC["?";"\"[
```

or

```
ESC[92;92pESC[63;63p
```

Display.sys

Display.sys is an installable device driver that supports code page switching for the console device.

To install *display.sys*, insert a command line of the following form in your *config.sys* file:

```
device=[drive:][path]display.sys con[:]=(type[,hwcp][,n,m])
```

The following list describes how each variable is used:

Option	Function
<i>type</i>	The display adapter in use. Valid values include MONO, CGA, EGA, and LCD. If <i>type</i> is omitted, <i>display.sys</i> checks the hardware to select the active display adapter type. EGA supports both EGA and VGA display adapters.
<i>hwcp</i>	The code page supported by the hardware. The following values are allowed: 437 (United States) 850 (Multilingual) 860 (Portugal) 863 (Canadian-French) 865 (Norway)
<i>n</i>	The number of additional code pages that can be supported (0-12). This number is dependent on the hardware. MONO and CGA do not support other fonts, so <i>n</i> must be 0. EGA can be 2. LCD can be 1.
<i>m</i>	The number of sub-fonts that are supported for each code page.

Example:

To install *display.sys* to support an EGA display adapter with a United States code page and the potential for two more code pages, include the following line in your *config.sys* file:

```
device=c:\dos\display.sys con:=(ega,437,2)
```

Installing Display.sys

The Display.sys options

Driver.sys

Driver.sys is an installable device driver that allows a physical disk drive to be referenced by a logical drive. A logical drive is a pointer to a physical disk drive in the system. It includes an optionally different set of parameters describing the disk drive to MS-DOS and is associated through a drive letter (for example, A).

If the intent is to *modify* the existing parameters of a logical drive, see the *drivparm* command in Appendix B, "How to Configure Your System".

Driver.sys is not for use with hard disks. If a logical drive letter is desired for a hard disk, see the *subst* command in Chapter 3, "MS-DOS Commands."

Installing Driver.sys

To install *driver.sys*, insert a command line of the following form in your *config.sys* file:

```
device=driver.sys /d:number [/c] [/f:factor] [/h:heads] [/s:sectors]
[/t:tracks]
```

The *driver.sys* device driver accepts the following switches:

The Driver.sys switches

Switch	Function
<i>/d:number</i>	Physical drive number, ranging from 0 to 127. The first physical floppy disk drive is number 0, and is referenced from the MS-DOS command line as drive A. Drive number 1 is the second physical floppy disk drive. Drive 2 is the third, which must be external.
<i>/c</i>	Indicates that the physical disk drive referenced has change-line support in the hardware. This means that the physical disk drive can sense that the drive door has been opened. If the door is open, the device driver will assume that the drive does not have a disk in it yet. If you specify this switch, MS-DOS assumes that the physical disk drive has the ability to support change-line error detection. See the documentation for your disk drive to see if it supports change-line.
<i>/f:factor</i>	<i>Factor</i> (also called the form factor) specifies the type of disk drive. It can be chosen from the list below, with the default being 2. 0 = 160/180 kilobytes, or 0 = 320/360 kilobytes 1 = 1.2 megabytes 2 = 720 kilobytes (3.5-inch disk) or others 7 = 1.44 megabytes (3.5-inch disk)

Switch	Function
<i>/h:heads</i>	Maximum head number, ranging from 1 to 99. The default value is 2.
<i>/s:sectors</i>	Number of sectors per track, ranging from 1 to 99. The default value is 9.
<i>/t:tracks</i>	Number of tracks per side on the block device ranging from 1 to 999. The default value is 80.

Examples:

To add an external 720 kilobyte drive to your computer, you would include the following line in the *config.sys* file:

```
device=driver.sys /d:2/f:2/h:2/s:9/t:80
```

This example assumes that you have *driver.sys* in the root directory of the boot drive.

To copy files from a 1.44 megabyte external disk drive to the same drive, put the identical line in your *config.sys* file. This will assign two logical drive letters to the same physical drive and MS-DOS will allow you to switch disks in the same drive during the copying process. For example:

```
device=driver.sys/d:2/f:7/h:2/s:18/t:80
device=driver.sys/d:2/f:7/h:2/s:18/t:80
```

If your machine has a 1.2 megabyte disk drive configured as drive A and a 360 kilobyte disk drive configured as drive B, *driver.sys* can be used to allow files to be copied using two disks of the same media type and using the same drive. In this example, *driver.sys* is used to assign a new logical drive to the 1.2 megabyte A drive (for example, drive D):

```
device=driver.sys/d:0/f:1/h:2/s:15/t:80
```

Now, a file can be copied from drive A to drive D using two 1.2 megabyte disks in the 1.2 megabyte physical drive, with MS-DOS prompting you to insert the applicable disk for the appropriate logical drive.

To create a new logical drive, having different parameters than the defaults associated with the physical drive, include the following line in the *config.sys* file:

```
device=driver.sys/d:1/f:2/h:2/s:9/t:80
```

This example illustrates a 720K physical drive configured as drive B. If the computer system does not fully support that drive type (in this case, 720K), MS-DOS will configure the logical drive B with the parameters corresponding to a 360K drive. Using drive B will lead to disk errors, but using the logical drive letter assigned with this command will reference the physical drive correctly, using the new parameters.

Note

If *driver.sys* is used to assign a logical drive with *different* parameters to a physical drive, it is assumed that the earlier logical drive's parameters are incorrect. Therefore, the drive letter corresponding to that earlier logical drive should not longer be used.

Printer.sys

Printer.sys is an installable device driver that supports code page switching for parallel ports LPT1, LPT2, and LPT3. (The port name PRN may be substituted for LPT1 to refer to the first parallel port.)

To install *printer.sys*, insert a command line of the following form in your *config.sys* file:

```
device=[drive:][path]printer.sys lptx= (type[,hwcp[,...]][,n])
```

The *printer.sys* device driver accepts the following options:

Option Function

type The printer in use. Valid types include any of the following printers or compatibles:

4201	IBM 4201 Proprinter family IBM 4202 Proprinter XL
4208	IBM 4207 or 4208 IBM Proprinter X24 or XL24
5202	IBM 5202 Quietwriter III

hwcp The code page supported by the hardware. The following values are allowed:

437 (United States)
850 (Multilingual)
860 (Portugal)
863 (Canadian-French)
865 (Norway)

n The number of additional code pages that can be supported. This number is dependent on the hardware.

Installing Printer.sys

The Printer.sys options

Ramdrive.sys

Ramdrive.sys is an installable device driver that lets you use a portion of your computer's memory as if it were a hard disk. This memory area is called a RAM disk and is sometimes referred to as a *virtual disk*.

RAM disks are much faster than hard disks because the information they contain is always loaded into memory. If your computer has extended memory installed (starting at the one megabyte boundary), or if you have an extended memory board that meets the Lotus®/Intel®/Microsoft® Expanded Memory Specification, you can use this extended memory for one or more RAM disks. Otherwise, *ramdrive.sys* places RAM disks in low memory.

Note

The command **device=ramdrive.sys** increases the size of MS-DOS resident in memory.

Installing Ramdrive.sys

To install *ramdrive.sys*, include a command line of the following form in your *config.sys* file:

device=ramdrive.sys [*disksize*] [*sectorsize*] [*entries*] [/e]

or

device=ramdrive.sys [*disksize*] [*sectorsize*] [*entries*] [/a]

The *ramdrive.sys* device driver accepts the following options:

Option	Function
The Ramdrive.sys options	
<i>disksize</i>	Specifies the disk size in kilobytes. The default size is 64K bytes; the minimum size, 16.
<i>sectorsize</i>	Specifies the sector size in bytes. The default size is 512 bytes. The following values are allowed: 128, 256, 512, and 1024 bytes.
<i>entries</i>	Specifies the number of root directory entries. The default value is 64; the minimum, 4; the maximum, 1024.
	Ramdrive.sys adjusts the value of <i>entries</i> to the nearest sector boundary. For example, if you specify a value of 25 when the sector size is 512 bytes, the 25 will be rounded up to 32, which is the next multiple of 16 (there are sixteen 32-byte directory entries in 512 bytes).

Option	Function
/e	Lets you use extended memory (above one mega-byte) as a RAM disk if it has been installed. If you use this switch, you cannot use the /a switch. It is recommended that you use the /e switch.
/a	Lets you use an expanded memory board that meets the Lotus/Intel/Microsoft Expanded Memory Specification for a RAM drive-if that board has been installed. If you use this switch, you cannot use the /e switch.

Note

When you reset or turn off the power on your computer, the information stored in RAM disks is lost.

Smartdrv.sys

Smartdrv.sys is a disk-caching program for computers that have a hard disk and expanded or extended memory. Disk-caching programs are designed to reduce the amount of time your computer spends reading data from your hard disk.

Smartdrv.sys is ideal for users who work with many applications and files at once. It is especially useful when running multiple standard applications that require swapping — copying applications to and from the hard disk to make room for all the applications in memory. Although *smartdrv.sys* can be used in conjunction with applications that use expanded or extended memory, it should not be used with other memory-disk or disk-caching programs.

Using Smartdrv.sys

To use *smartdrv.sys* with your computer, you need the following:

- An IBM PS/2, IBM PC-XT, or IBM PC—compatible with a hard disk.
- An expanded memory board or an extended memory board.

Installing Smartdrv.sys

Smartdrv.sys works best with expanded memory, but it will also provide good results with extended memory. To determine which type of memory to use with *smartdrv.sys*, follow these rules: if you have only expanded memory, use expanded; if you have only extended memory, use extended; if you have a memory card that can be set up for use as expanded or extended memory, use extended memory for *smartdrv.sys* and leave expanded memory for other expanded-memory applications.

To install *smartdrv.sys*, include a command line of the following form in your *config.sys* file:

```
device=[d:][path]smartdrv.sys [size][/a]
```

The *smartdrv.sys* device driver accepts the following options:

The Smartdrv.sys options

Option	Function
<i>size</i>	The amount of memory you want <i>smartdrv.sys</i> to have. If you don't specify an amount of memory, <i>smartdrv.sys</i> uses 256K if it is installed in extended memory, or all of expanded memory if it is installed in expanded memory. If you plan to run a standard application that uses expanded or extended memory, you should specify a size that leaves enough memory for that application.

Option	Function
/a	Use this switch if you have expanded memory or if you want to use your extended memory board as expanded memory.

Examples:

The following command line gives *smartdrv.sys* 1024K of expanded memory:

```
device=smartdrv.sys 1024/a
```

The following command line gives *smartdrv.sys* all the expanded memory:

```
device=smartdrv.sys /a
```


Appendix D

Configuring Your Hard Disk (Fdisk)

Introduction

Hard disks can be divided into one to four separate sections, called *partitions*. Partitions separate your hard disk into individual areas, and each partition may contain a different operating system.

To prepare your hard disk for the MS-DOS operating system, you must create a partition for MS-DOS, called a DOS partition. You can create a DOS partition on your hard disk by using a menu-driven utility called **fdisk**. You must use **fdisk** if you want to do one of the following:

- Create a primary MS-DOS partition
- Create an extended DOS partition
- Create a logical DOS drive
- Set the active partition
- Delete a DOS partition or logical drive
- Display partition information
- Review or modify the configuration of another hard disk on your computer

Warning

Reconfiguring your disk with **fdisk** destroys all existing files. Be sure to have a backup of all files on your disk *before* you create an MS-DOS partition with **fdisk**.

Checking for a Configured and Formatted Disk

Has Fdisk been run already?

Many computer stores configure hard disk computers for MS-DOS, so you may not need to use **fdisk**. They may also format your hard disk to start MS-DOS when you turn the power on. To find out whether this has been done, do the following:

- 1 First, try to start MS-DOS from your hard disk.
 - If it starts, your hard disk is both configured and formatted, and the MS-DOS system files are on the disk.
 - If MS-DOS does not start, your disk is not formatted to start MS-DOS, but may have been configured.
- 2 If MS-DOS did not start, check to see if the disk has been configured with **fdisk**. Place the MS-DOS master floppy disk in drive A and press **CONTROL+ALT+DELETE** to start MS-DOS. Then run **fdisk** and select the "Display Partition Data" to see if any MS-DOS partitions exist.
 - If any do exist, your disk has been configured.
 - If no partitions exist, follow the instructions in this appendix to configure your disk.
- 3 After your hard disk is configured, be sure to format your disk with the MS-DOS **format /s** command before you copy files onto the disk. Otherwise, your files will be unreadable.

How to Start Fdisk

Starting Fdisk

The **fdisk** utility is easy to use because it uses menus to lead you through each procedure. To start **fdisk**, follow these steps:

- 1 Place the MS-DOS disk in drive A.
- 2 Turn on your computer to start MS-DOS.
- 3 Type the following command and press the **ENTER** key:

Fdisk

Fdisk main menu

In response, **fdisk** displays its main menu on your screen. This menu lists five choices. If your computer has only one hard disk, Choice 5 will not appear on your screen.

Disk Options

Current fixed disk drive: 1

Choose one of the following:

1. Create DOS Partition or Logical DOS drive
2. Set Active Partition
3. Delete DOS Partition or Logical DOS drive
4. Display Partition Data
5. Select Next Fixed Disk Drive

Enter choice: [1]

Press ESC to return to DOS

The following sections describe each of these options, and show the menus and other information they display. To return to MS-DOS from the main menu, just press the `esc` key. You can also use the `esc` key to return to the main menu from any of the `fdisk` menus.

Most of the `fdisk` menus displays a default value. To choose the default value, press the `ENTER` key. To choose another value, just type the value you want, and press the `ENTER` key.

How to Create a DOS Partition

If you choose the first option on the main menu, and if your hard disk is not yet completely partitioned, `fdisk` displays a screen like the following. If no extended partitions exist, the third option is not displayed:

Fdisk menu: option 1

Create DOS Partition

Current fixed disk drive: 1

1. Create Primary DOS Partition
2. Create Extended DOS Partition
3. Create Logical DOS Drive(s) in
the Extended DOS Partition

Enter choice: [1]

Press ESC to return to Fdisk Options

Selection 1: Create Primary DOS Partition

You must create a primary MS-DOS partition first before you can create any extended MS-DOS partitions on your disk. In most cases, you will need only one MS-DOS partition for your entire disk.

Creating a primary MS-DOS partition

To create a primary MS-DOS partition, press the **ENTER** key to accept the default selection (1).

The Create Primary DOS Partition menu appears next:

```
Create Primary DOS Partition
Current fixed disk drive: 1

Do you wish to use the maximum size for a
DOS partition and make the DOS partition
active (Y/N).....? [Y]

Press ESC to return to Fdisk Options
```

Creating a single MS-DOS partition

If you use your entire hard disk for MS-DOS, you will use the **fdisk** program only once to create the primary MS-DOS partition. If you want to use the entire hard disk (up to 2 gigabytes) for MS-DOS, press the **ENTER** key to accept the default selection (Y).

Fdisk then displays the following message:

```
System will now restart
Insert DOS diskette in drive A:
Press any key when ready . . .
```

Put your MS-DOS disk in drive A and press any key to restart MS-DOS.

Formatting your hard disk to start MS-DOS

Now that you have created your MS-DOS partition, you must format your hard disk so that MS-DOS can use it. If you want to start MS-DOS from your hard disk, remember to use the **/s** switch with the **format** command. For example, if you are formatting the disk in drive C and want to start MS-DOS from that disk, type the following command:

```
format c: /s
```

(For more information about the **format** command, see Chapter 3, "MS-DOS Commands".)

Creating More than One MS-DOS Partition

You may choose to create a primary MS-DOS partition smaller than the maximum size. To do this, type N (for No) in response to the question on the first Create Primary DOS Partition menu. Fdisk displays a second Create Primary DOS Partition menu like the following. From this menu, you can specify the size of the primary MS-DOS partition:

```

Create Primary DOS Partition

Current fixed disk drive: 1

Total disk space is 42 Mbytes
(1Mbyte = 1048576 bytes).

Maximum space available for partition
is 42 Mbytes (100%).

Enter partition size in Mbytes or percent
of disk space (%) to create a Primary
DOS partition..... [ 42]

Press ESC to return to Fdisk Options

```

The space available on your hard disk is measured in *megabytes*. This menu shows the total number of megabytes available for a hard disk partition, and prompts you to enter the size of your new partition. The default size for the partition is the maximum available space on the hard disk. Press the ENTER key if you want the default size. Otherwise, type the size (in megabytes), or enter the percentage of the disk to use followed by a percent (%) sign. After entering the amount you want for the partition, press the ENTER key.

Any part of the disk that you do not use for the primary MS-DOS partition may be used for an extended MS-DOS partition.

Selection 2: Create Extended DOS Partition

You can use **fdisk** to create an extended partition if you want to designate one or more logical drives for the disk.

Creating an extended DOS partition

To select Create Extended DOS Partition, type 2, then press the **ENTER** key. In response, **fdisk** displays a menu like this one:

```

Create Extended DOS Partition

Current fixed disk drive: 1

Partition Status Type Size in Mbytes Percentage
                                of Disk Used
C:1      PRI    DOS      21      50

Total disk space is 42 Mbytes (1Mbyte = 1048576 byte:
Maximum space available for partition
is 21 Mbytes (50%).
Enter partition size in Mbytes or percent of disk sp:
to create an Extended DOS partition..... [ 21

Press ESC to return to Fdisk Options
  
```

This menu shows the total number of megabytes available for an extended partition. The default for the partition size is the maximum available space on the hard disk. Press the **ENTER** key if you want the default. Otherwise, type the size (in megabytes), or enter the percentage of the disk to be used followed by a percent (%) sign. After entering the amount you want for the partition, press the **ENTER** key.

Note

If **fdisk** finds any defective tracks at the start of the partition, it adjusts the partition boundaries to avoid those bad tracks.

Selection 3: Create Logical Drive in the Extended DOS Partition

Creating logical drives in extended partitions

When you have created an extended partition, you must specify one or more drive letters for that area of the disk. After you create an extended partition and if you choose option 3 from the Create DOS Partition menu, **fdisk** displays a menu similar to the following:

```

Create Logical DOS Drive(s) in the Extended
DOS partition

Total partition space is 21 Mbytes (1 Mbyte =
1048576
bytes). Maximum space available for logical
drive is 21 Mbytes (100%).

Enter logical drive size..... [ 21]

Press ESC to return to Fdisk Options
  
```

You may designate the entire partition as one logical drive, or divide it into two or more logical drives. For example, if you want to segregate a particular application and its data files to their own drive, you may want to create a second logical drive on the partition.

Because you cannot use an MS-DOS extended partition without a drive letter, **fdisk** continues to prompt you for logical disk drive information until the whole partition has been assigned to a logical drive.

When the entire partition is assigned to logical drives, **fdisk** displays this message:

```
All available space in the Extended DOS
partition is assigned to logical drives.
```

Press **ESC** to return to the main **fdisk** menu. From there, you can restart MS-DOS, or select another option.

How to Set the Active Partition

If you choose the second option on the main menu, **fdisk** displays a screen showing the status of each partition on your hard disk. The *active partition*, indicated by a status of **A**, contains the operating system and files you access when you turn on or reset your computer. If you have created a partition on your disk with another operating system, this menu allows you to make that partition the active partition. Only one partition is active at a time; the others are not active.

For example, if you have both XENIX and DOS partitions on your disk, the Set Active Partition menu might look like this:

```
Set Active Partition
```

```
Current fixed disk drive: 1
```

Partition	Status	Type	Size in Mbytes	Percentage of Disk Used
C: 1		non DOS	21	50
2	A	PRI DOS	21	50

```
Total disk space is 42 megabytes.
```

```
Enter the number of the partition you
want to make active.....: [ 1]
```

```
Press ESC to return to Fdisk Options
```

Fdisk menu: option 2

Setting the active partition

Type the number of the partition that you want to activate, and press the **ENTER** key. The default setting is the current active partition number.

If your hard disk contains only MS-DOS partitions, only the primary DOS partition can be active. If you try to activate an extended DOS partition, **fdisk** displays the following message:

```
Partition selected (3) is not startable,  
active partition not changed.
```

```
Press ESC to return to FDISK Options.
```

How to Delete a DOS Partition

Fdisk menu: option 3

If you choose the third option on the main menu, **fdisk** displays the following menu, which asks you to identify whether the partition you want to delete is a primary or extended DOS partition:

```
Delete DOS Partition
```

```
Current fixed disk drive: 1
```

1. Delete Primary DOS Partition
2. Delete Extended DOS Partition
3. Delete Logical DOS Drive(s) in the Extended DOS Partition

```
Enter choice: [ ]
```

```
Press ESC to return to Fdisk Options
```

Deleting a DOS partition

Type the number of the selection you want and press the **ENTER** key. The next menu, whether for a primary or extended DOS partition, shows the status of that partition. When you delete a DOS partition, **fdisk** deletes the partition boundaries and any data that existed in that partition. Once you delete the partition, *you cannot recover the data that was on it.*

Note

You cannot use **fdisk** to delete a non-DOS partition. Instead, to continue using MS-DOS after you have deleted the DOS partition, you must put an MS-DOS program disk into drive A. To start a different operating system in another partition of your hard disk, you must change the active partition to that number *before* you delete the DOS partition.

Selection 1: Delete Primary DOS Partition

If you choose to delete the Primary DOS Partition, you must first delete all the extended DOS partitions on the drive.

Deleting the primary DOS partition

The Delete Primary DOS Partition menu will look similar to this:

```
Delete Primary DOS Partition
```

```
Current fixed disk drive: 1
```

```
Partition Status Type Size in Mbytes Percentage
                                of Disk Used
C: 1          A     PRI DOS    21          50
```

```
Total disk space is 42 Mbytes (1 Mbyte = 1048576
bytes).
```

```
Warning! Data in Primary DOS
partition will be lost. Do you wish
to continue.....? [N]
```

```
Press ESC to return to Fdisk Options
```

If you do *not* want to delete the primary DOS partition, press the ENTER key to accept the default value (N).

To delete the primary DOS partition,

- 1 Type Y.
- 2 Press the ENTER key.

Selection 2: Delete Extended DOS Partition

If you choose to delete an extended partition, you must first delete the logical drives associated with that partition.

Deleting an extended DOS partition

Selection 3: Delete Logical Drive in the Extended DOS Partition

Deleting a logical drive

To delete a logical drive, type 3 to select the option Delete Logical DOS Drive(s) in the Extended DOS Partition from the Delete DOS Partition menu. Then press the **ENTER** key. **Fdisk** displays a menu like the following:

```

Delete Logical DOS Drive(s)

Drv Volume Label Mbytes System Usage
D: 19 DOS 90%
E: 2 DOS 10%

Total extended partition space is 21 Mbytes
(1 Mbyte = 1048576 bytes).

Warning! Data in the logical DOS drive
will be lost. What drive do you wish
to delete.....? [ ]

Press ESC to return to Fdisk Options

```

Type the letter of the drive you want to delete, and press **ENTER**. **Fdisk** displays this message:

```

Enter Volume Label [ ]

Are you sure.....? [N]

```

If this logical drive contains valuable data that you have not backed up, press **ENTER**. This stops **fdisk** from deleting the logical drive.

Note

Be sure to back up all files you will need from the logical drive *before* you delete the drive. When **fdisk** deletes a logical drive or partition, the data is destroyed.

If you are sure you want to delete the drive, type **Y** (for Yes).

How to Display Partition Data

Fdisk menu: option 4

If you choose the fourth option on the main menu, **fdisk** displays a menu that contains information about each of the partitions on your hard disk.

For example, the Display Partition Information menu might look like this:

Display Partition Information

Current fixed disk drive: 1

Partition	Status	Type	Size in Mbytes	Percentage of Disk Used
C: 1	A	PRI DOS	21	50
2		EXT DOS	21	50

Total disk space is 42 Mbytes
(1 Mbyte = 1048576 bytes).

The Extended DOS partition contains
logical DOS drives. Do you want to
display logical drive information? [Y]

Press ESC to return to Fdisk Options

This information screen identifies the partitions on your disk. It shows each partition's number, status, and type; its size in megabytes; and percentage of disk used.

If you have an extended partition, **fdisk** asks if you want to see information about that partition's logical drives. Type *Y* and press ENTER to display a screen like the following:

Display Logical DOS Drive(s) Information

Drv	Volume	Label	Mbytes	System	Usage
D:	19	DOS	90%		
E:	2	DOS	10%		

Total Extended DOS Partition size is 21 Mbytes
(1 Mbyte = 1048576 bytes)

Press ESC to return to Fdisk Options

Press the **esc** key to return to the main menu.

How to Select the Next Fixed Disk Drive

Fdisk menu: option 5

This option appears on the **fdisk** main menu only if you have more than one hard disk attached to your computer. If you choose this option, **fdisk** changes the current disk drive to the next drive.

Selecting the next fixed disk drive

For example, if the current disk drive is drive C, and if you choose option 5 on the main menu, **fdisk** changes the current disk drive to drive D. You could then choose any of the **fdisk** options (1–4) to prepare the second fixed disk for MS-DOS. Or, you could select option 5 once again to select the next drive. For example, if there is not a third fixed disk, **fdisk** changes the current disk drive from D back to C.

After you have selected the next drive, **fdisk** displays the main menu again. Notice that near the top of the screen, there is a line that looks something like this:

```
Current fixed disk drive: 2
```

The activity you select will be performed on the disk shown in this line.

Appendix E

How to Use Code Pages

Introduction

MS-DOS 4.0 provides national language support through the use of language-specific code pages. If you live in, or work with, a country other than the United States, you may choose to use the MS-DOS commands that support code-page switching

A *code page* is a table that defines the character set you are using. Each *character set* contains 256 entries specific to a country or language. The characters are translated from the code page table and used by your keyboard, screen, and printer. An example of a character set is the set of letters, numbers, and symbols (such as accent marks) used by French-Canadians. When the character set is put in a table for use by MS-DOS, it becomes the Canadian-French code page.

There are two types of code pages, hardware and prepared. A *hardware* code page is built into a device. For example, a printer manufactured for use in Portugal has a Portuguese hardware code page in it. Many devices can use only their own hardware code page.

Prepared code pages are provided in code-page information (.cpi) files in your software. MS-DOS includes the following prepared code pages:

- 437—United States code page.
- 850—Multilingual code page. This code page includes all characters for most languages of European, North American, and South American countries.
- 860—Portuguese code page.
- 863—Canadian-French code page.
- 865—Nordic code page. This code page includes all characters for the Norwegian and Danish languages.

What Is a code page?

Devices that can use multiple languages

The following devices can use prepared code pages to switch from one language to another.:

Enhanced Graphics Adapter (EGA) or compatible
 IBM Personal System/2 display adapter or compatible
 IBM PC Convertible LCD or compatible
 IBM Proprinter Model 4201 or compatible
 IBM Proprinter Model 4208 or compatible
 IBM Quietwriter III Printer Model 5202 or compatible

For example, if you have an EGA, you can switch between the United States code page and the Multilingual code page by entering the appropriate MS-DOS commands. With prepared code pages, a single system can support multiple national languages.

National Language Support Codes

When you use commands to set up your system for a national language, MS-DOS checks to make sure the screen, printer, and keyboard codes you request work together. The valid combinations of prepared code pages, country codes, and keyboard codes are defined in the *country.sys* and *keyboard.sys* files. The following combinations are supported:

Country or language	Country code	Prepared code pages	Keyb code
United States	001	437,850	us
Canadian-French	002	863,850	cf
Latin America	003	437,850	la
Netherlands	031	437,850	nl
Belgium	032	437,850	be
France	033	437,850	fr
Spain	034	437,850	sp
Italy	039	437,850	it
Switzerland	041	437,850	sf,sg
United Kingdom	044	437,850	uk
Denmark	045	865,850	dk
Sweden	046	437,850	sv

Country or language	Country code	Prepared code pages	Keyb code
Norway	047	865,850	no
Germany	049	437,850	gr
English (International)	061	437,850	—
Portugal	351	860,850	po
Finland	358	437,850	su
Israel	972	437	—
Japan	081		ja
Korea	082		ko
Peoples Republic of China	086		ch
Taiwan	088		tn

Notes

- Both Swiss-French and Swiss-German use country code 041.
- Country codes 358 and 972 assume United States code page 437, but include country-specific date and time conventions.

For example, if you use country code 002 (Canadian-French), you can use prepared code pages 863 and 850 and the CF (Canadian-French) keyboard code. You could not use the US (United States) keyboard code. If you enter a country code 002 with a US keyboard code, MS-DOS displays an error message when you restart the system.

How to Use Code Pages

MS-DOS uses several commands to support a national language with a code page. To set up your system to support a character set, you need to do the following:

In your `config.sys` file:

- Use the **country** configuration command to control country-specific characteristics such as the time format, date format, currency symbol, and character-sorting sequence.
- Use a **device** configuration command to tell MS-DOS the hardware code page in a device and to allocate buffers for prepared code pages.

Setting the system code page

Note

Appendix B, "How to Configure Your System", explains each configuration command. Examples of *config.sys* commands are shown later in this chapter. Also, remember that when you change your *config.sys* file, you must restart MS-DOS to enable the new settings.

In your *autoexec.bat* file:

- Use an *nlstunc* command to load the memory-resident national support functions. If you forget to issue the *nlstunc* command, MS-DOS will not allow you to specify code pages or keyboard codes.
- Use a *mode cp prepare* command to prepare code pages for each device that supports code-page switching.
- Use a *keyb* command to select the initial keyboard layout.
- Use a *chcp* command to select the code pages for all prepared devices. MS-DOS automatically prepares two system code pages and selects the primary code page for your country automatically. If you want to use the other code page prepared for your country, you can use the *chcp* command again.

Note

For an explanation of each command, see Chapter 3, "MS-DOS Commands". Examples of how to use the commands are shown later in this chapter.

Several examples, using different hardware and prepared code pages, are provided on the following pages. The examples show you how to set up your system to support national languages. The examples assume all MS-DOS files are in the directory \dos on drive C.

Setting country and keyboard codes

The first example uses only a hardware code page; code-page switching is not used. The commands tell MS-DOS to accept the country information for France (code page 437) and loads the French keyboard program. This example covers most European countries.

In your *config.sys* file, type the following:

```
country=033 , , c:\dos\country.sys
```

In your *autoexec.bat* file, type the following:

```
keyb fr , , c:\dos\keyboard.sys
```

The next example uses two code pages—one hardware and one prepared. In this instance, the system you're setting up uses the German code page provided in the EGA and the United States code page provided by MS-DOS.

Using hardware and prepared code pages

In your *config.sys* file, type the following:

```
country=049 , , c:\dos\country.sys
device=c:\dos\display.sys con=(ega, 437, 1)
```

These *config.sys* statements tell MS-DOS that the hardware code page (437) in the enhanced graphics adapter should be used and that you are allocating space for one prepared code page.

In your *autoexec.bat* file, type the following:

```
nlsfunc
mode con codepage
prepare=(850)c:\dos\ega.cpi)
keyb gr , , c:\dos\keyboard.sys
chcp 437
```

These *autoexec.bat* commands do the following:

- Load memory-resident, national support functions.
- Load prepared code page 850 into memory. They get the code page from the *ega.cpi* file.
- Load the German (gr) keyboard program.
- Start the system with the United States (437) code page.

How to Set Device Code Pages

The following example uses two prepared code pages—no hardware code pages are used—to set up the system to support Denmark/Norway and the Multilingual national languages.

Setting screen code pages

In your *config.sys* file, type the following:

```
country=045 , , c:\dos\country.sys
device=c:\dos\display.sys con=(ega, , 2)
```

These *config.sys* commands tell MS-DOS that the EGA's hardware code page is not used and that you are allocating space for two prepared code pages.

In your *autoexec.bat* file, type the following:

```
nlsfunc
mode con codepage prepare=((865, 850)
c:\dos\ega.cpi)
keyb dk , , c:\dos\keyboard.sys
chcp 865
```

These *autoexec.bat* commands do the following:

- Load memory-resident, national support functions.
- Load prepared code pages 865 and 850 into memory. They get the code pages from the *ega.cpi* file.
- Load the Denmark (dk) keyboard program.
- Start the system with the Denmark/Norway (865) code page.

Setting parallel printer code pages

The last example also uses two prepared code pages. And it assumes you have an IBM Proprinter model 4201 connected to LPT1. The system is set up to support Denmark/Norway and the Multilingual national languages.

In your *config.sys* file, type the following:

```
country=045 , , c:\dos\country.sys
device=c:\dos\display.sys con=(ega,,2)
device=c:\dos\printer.sys lpt1=(4201,,2)
```

These *config.sys* commands tell MS-DOS that the hardware code page is not used and that you are allocating space for two prepared code pages in each device.

In your *autoexec.bat* file, type the following:

```
nlsfunc
mode con codepage
prepare=(865,850) c:\dos\ega.cpi
mode con codepage
prepare=(865,850) c:\dos\4201.cpi
keyb dk, , c:\dos\keyboard.sys
chcp 865
```

These *autoexec.bat* commands do the following:

- Load memory-resident, national support functions.
- Load prepared code pages 865 and 850 into memory. For the EGA, they get the code pages from the *ega.cpi* file. For the printer, they get the code pages from the *4201.cpi* file.
- Load the Denmark (dk) keyboard program.
- Start both devices with the Denmark/Norway (865) code page.

How to Switch Between Code Pages

After using the *config.sys* and *autoexec.bat* file commands to set up your system for multiple national languages, use the following command to change to a different code page on all devices or for a single prepared device:

chcp *code page*

or

mode *device* **cp** *select=code page*

where:

code page is one of the pages set up in *autoexec.bat*. *device* is CON or LPT1.

How to List Current Code Pages

You can list the current prepared and selected code pages for your console screen or a parallel printer by using the **mode** command in the following form:

mode *device* **codepage**

For example, to display the current code pages for your console screen device, type the following:

```
mode con codepage
```

MS-DOS displays a message similar to this one:

```
Active codepage for device CON is 437
hardware codepages:
  Codepage 850
prepared codepages:
  Codepage 437
  Codepage 850
  Codepage not prepared
  Codepage not prepared
MODE Status Codepage function completed
```

**Displaying current device
code pages**

How to Refresh Lost Code Pages

It is possible for prepared code pages to be lost due to hardware errors or other reasons. For example, if you prepared code pages for your printer, and then turned off the printer, the current code page may be lost. You can use the **refresh** keyword with the **mode** command to restore the lost code page.

Refreshing lost code pages

To illustrate, suppose you had selected the Canadian-French code page (863) as the active code page for your console screen (CON). But, because of a hardware error, the active code page was lost. You could type the following commands to reinstate the active code pages for your screen:

```
mode con codepage prepare=(863)ega.cpi)

mode con cp refresh
```


Appendix F

MS-DOS Message Directory

MS-DOS Messages

There are three types of messages that you could see on your screen:

- MS-DOS utility messages
- MS-DOS device error messages
- Application program messages

MS-DOS utility and device error messages are listed in this appendix. For instructions about error messages related to non-MS-DOS software, see your application's documentation.

If a disk or device error occurs at any time during a command or program, MS-DOS displays an error message and includes this prompt:

Abort, Ignore, Retry, Fail?_

MS-DOS waits for you to type one of the following responses:

- A** Abort. End the program requesting the disk read or write.
- I** Ignore. Ignore the bad sector and pretend the error did not occur. This may result in lost data.
- R** Retry. Repeat the operation. You should use this response when you have corrected the error (for example, with Not ready or Write protect errors).
- F** Fail. This causes the current MS-DOS system operation to end (fail) and the application to continue.

Note

For some floppy disk device errors, the Ignore option will not be displayed. This prompt instead will read

Abort, Retry, Fail?_

Usually, you will want to recover by first typing R (to try again). If the second attempt fails, type A to terminate the process.

This section describes MS-DOS messages, their causes, and how to correct them. It identifies the list of sources in brackets ([]) for each message.

10 Mismatches - ending compare

[Comp]

Comp assumes the files are too different after 10 mismatches and stops.

80386 XMA Emulator Installed

[Xmaem.sys]

This is an informational message indicating that the XMA board emulator has been installed.

**80386 XMA Emulator not Installed
Insufficient memory for requested parameter value.**

[Xmaem.sys]

*You have requested more pages be allocated than can fit in the amount of extended memory available.
Reduce the number of pages requested.*

**80386 XMA Emulator not Installed
Protected mode already in use.**

[Xmaem.sys]

The emulator has found the CPU already in protected mode. The emulator was not loaded as a result.

**80386 XMA Emulator not Installed
Parameter value less than 64 page minimum required.**

[Xmaem.sys]

You requested less than 64 pages be allocated. At least 64 pages must be present.

**80386 XMA Emulator not Installed
This system unit is not supported.
The 80386 XMA Emulator requires a personal computer with an 80386 CPU.**

[Xmaem.sys]

You are trying to load the emulator on a computer which doesn't have an 80386.

80386 XMA Emulator Version 1.10

[Xmaem.sys]

This is an informational message.

Abort edit (Y/N)?

[Edlin]

MS-DOS displays this message when you choose the Edlin Q (quit) command. The Q command exits the editing session without saving any editing changes.

Type Y (for Yes) or N (for No).

Access denied

[Attrib][Find][Print][Replace][Xcopy]

You tried to replace a write-protected, read-only, or locked file.

Active Code Page: xxx

[Chcp]

xxx is the code page currently being used by the system.

Active Code Page for device ddd is xxx

[Mode]

xxx is the code page currently being used by the device ddd.

Active Code Page not available from con device

[Keyb]

The code page that the system is currently using is not supported on the console (screen) you are using.

Adapter error

[Xma2ems.sys]

Hardware problems were encountered during startup. The LIM 4 driver was not installed.

Add filename? (Y/N)

[Replace]

Replace displays this prompt if you specify the /w switch.

Type Y (for Yes) if you want to add the file to the disk, or N (for No) if you do not want to add the file.

Adding filename

[Replace]

Replace displays this prompt to let you know that it is adding this file to your disk.

All available space in the Extended DOS Partition is assigned to logical drives

[Fdisk]

There is no more room for logical drives in the extended partition.

All files canceled by operator

[Print]

MS-DOS displays this message when you specify the /t switch with the Print command.

All files in directory will be deleted!

Are you sure (Y/N)?

[MS-DOS]

*MS-DOS displays this message if you try to delete all files in the working directory by using the *.* wildcard.*

Type Y (for Yes) to delete all the files, or N (for No).

All logical drives deleted in the Extended DOS Partition

[Fdisk]

Any logical drives previously associated with the extended DOS partition on your disk are now removed.

Allocation error, size adjusted

[Chkdsk]

The size of the file indicated in the directory was not consistent with the amount of data actually allocated to the file. The file was truncated to match the amount of data allocated.

All specified file(s) are contiguous

[Chkdsk]

All files are written sequentially on the disk.

To correct this error automatically, specify the chkdsk /f switch.

**An error occurred while installing DOS
Press Enter to continue, or Esc to exit SELECT**

[Select]

This is an unexpected error and you should respond appropriately to the prompt.

ANSI.SYS must be installed to perform requested function

[Mode]

The screen function requested requires ansi.sys be loaded.

APPEND already Installed**[Append]**

You have already used the Append command once since you turned on your computer. Now you are trying to use either the /x or /e switch with this command. These switches are only valid the first time you type the Append command.

If you want to change the append switch, reboot your computer. Then type the append command with the switch you want to use. Otherwise, use the append command without these switches. For more information about the append command, see Chapter 3, MS-DOS Commands.

APPEND/ASSIGN Conflict**[Append]**

You cannot use the Append command on an assigned drive.

Cancel the drive assignment before using the append command with this drive again.

Attempting to recover allocation unit X**[Format]**

Format found a bad allocation unit while trying to format the disk.

/B Invalid with a black and white printer**[Graphics]**

Since you cannot print a background color on a black and white printer, you cannot use the /B switch here.

***** Backing up files to drive x: *****

Diskette Number: n

[Backup]

Backup displays this message while backing up files to the specified drive.

Be sure to label backup disks with the appropriate backup disk number for use in restoring them later.

Bad command or file name**[MS-DOS]**

The command cannot find the program you asked it to run.

Check to see that you typed the command line properly, and that the file or command is on the disk or in the command path.

Bad or missing d:path SMARTDRV.SYS

[Smartdrv.sys]

The smartdrv.sys file is not in the drive or directory shown in the SMARTDrive command line.

Edit your config.sys file and type the correct drive and pathname in the smartdrv.sys line.

Bad or missing Command Interpreter

[MS-DOS]

MS-DOS cannot find the command.com file on the disk; either the file is missing from the root directory, or the file is invalid. You also receive this message if command.com has been moved from the directory it was originally in when you started MS-DOS.

Either restart the system with a disk that contains the command.com file, or copy the command.com file from your backup MS-DOS master disk onto the disk used to start MS-DOS.

Bad or missing filename

[MS-DOS]

You specified a device incorrectly in the config.sys file.

Check the accuracy of the device command in the config.sys file.

Bad or Missing Keyboard definition file

[Keyb]

MS-DOS cannot find the Keybxx file that you specified with the Keyb command.

Check to see that the file you specified exists on the disk. Also check to see that your path includes the directory in which this file resides. Then, retype the command. If you get this message again, the keyboard.sys or keyb.com file may be corrupted.

Bad Partition Table

[Format]

This message means that there is no DOS partition on the hard disk.

You must run fdisk to create a DOS partition on your hard disk.

Bad unit error reading drive x:

[MS-DOS device error]

Invalid subunit numbers were passed to the device driver.

Batch file missing

[Command]

A batch file name was referenced that could not be found.

Baud rate required

[Mode]

Mode COMx commands must have at least a baud rate specified for Mode to initialize the COM port.

BREAK is off (or on)

[MS-DOS]

This message tells you the current setting of Break.

Cannot CHDIR to path - tree past this point not processed

[Chkdsk]

Chkdsk is checking the structure of the directory and is unable to go to the specified directory. All subdirectories underneath this directory will not be verified.

To correct this error automatically, specify the chkdsk /f switch.

Cannot CHDIR to root

[Chkdsk]

Chkdsk is checking the tree structure of the directory and is unable to return to the root directory. Chkdsk is not able to continue checking the remaining subdirectories.

Try to restart MS-DOS. If this error persists, the disk is unusable.

Cannot Chkdsk a Network drive

[Chkdsk]

You cannot check drives that are redirected over the network.

Cannot Chkdsk a SUBSTed or ASSIGNED drive

[Chkdsk]

You cannot check drives that have been substituted or assigned.

Cannot create a zero size partition

[Fdisk]

You are trying to create a partition with a size of 0 cylinders. You must allocate a minimum of 1 cylinder to any partition you create.

Cannot create extended DOS partition without primary DOS partition on disk 1

[Fdisk]

You are trying to create an extended DOS partition, but your first hard disk does not contain a primary DOS partition.

First, create the primary DOS partition on your first hard disk. Then, if you have more room on that disk, or if you have a second hard disk, you can create an extended DOS partition.

Cannot create Logical DOS drive without an Extended DOS Partition on the current drive

[Fdisk]

You must create an extended DOS partition before any logical drives may be created.

Cannot delete Extended DOS Partition while logical drives exist

[Fdisk]

Your disk has one or more logical drives assigned to it. These must be removed before you can delete an extended DOS partition.

Delete all logical drives by using fdisk. Then delete the extended DOS partition.

Cannot DISKCOMP to or from an ASSIGNED or SUBSTed drive

[Diskcomp]

One of the drives that you specified is a drive that you created using the Assign or Subst command.

Cannot DISKCOMP to or from a network drive

[Diskcomp]

You cannot compare disks on drives that have been redirected over the network.

Cannot DISKCOPY to or from an ASSIGNED or SUBSTed drive

[Diskcopy]

One of the specified drives was created with the Assign or Subst command.

Cannot DISKCOPY to or from a network drive**[Diskcopy]**

You cannot copy disks to or from drives that have been redirected over the network.

Cannot do binary reads from a device**[Copy]**

The copy cannot be done in binary mode when you are copying from a device.

You should either not use the /b switch, or you should use the /a switch to specify an ASCII copy.

Cannot edit .BAK file—rename file**[Edlin]**

You attempted to edit a file that had a filename extension of .bak (a backup copy created by Edlin).

If you must edit a file that has an extension of .bak, you must either rename or copy the file and give it a different extension.

Cannot exec BASICA.COM**[MS-DOS]**

BASICA cannot be executed by MS-DOS.

Check to see that the basica.com file is on the disk you are using. If basica.com is not in your working directory, make sure that the path command points to the directory in which it is located. Try executing BASICA again. If you get the same message, the file itself may be bad. Try restoring basica.com from backup.

Cannot find adapter**[Xma2ems.sys]**

During initialization, the LIM 4 driver couldn't find an XMA memory adaptor in your system.

Cannot find FORMAT.COM

[Backup] You tried to back up to an unformatted disk and Backup couldn't find the Format command.

Cannot find GRAPHICS profile**[Graphics]**

You must give the path to the graphics.pro file or MS-DOS will look in the current directory.

Cannot find System Files**[Format]**

You tried to use a drive that did not have the system files in the root directory.

Cannot find XMA Emulator device driver

[Xma2ems.sys]

This is an informational message.

When you create your config.sys file, be sure to specify the xmaem.sys file before the xma2ems.sys file.

Cannot format an ASSIGNED or SUBSTed drive

[Format]

You attempted to format a drive currently mapped to another drive by the Assign or Subst command.

Run assign or subst again and clear all drive assignments.

Cannot FORMAT a Network drive

[Format]

You cannot format drives that are redirected over the network.

Cannot FORMAT nonremovable drive x:

[Backup]

You are trying to back up files with the /f switch. MS-DOS will not allow you to format the target disk specified.

Be sure you want to back up files to a hard disk. If you do, you must use a hard disk that is formatted already.

Cannot JOIN a Network drive

[Join]

You cannot join drives that are redirected over the network.

Cannot LABEL a Network drive

[Label]

You cannot label a drive that is shared on a network server station.

Cannot LABEL a SUBSTed or ASSIGNED drive

[Label]

You cannot label a drive if it has been substituted with the Subst command or assigned with the Assign command.

Check the command line to be sure you specified a valid filename.

Cannot load COMMAND, system halted

[MS-DOS]

MS-DOS cannot reload the command processor.

Reboot MS-DOS on your system.

Cannot perform a cyclic copy

[Xcopy]

When you are using the /s switch, you may not specify a target that is a subdirectory of the source.

Cannot read file allocation table

[Recover]

The file allocation on your disk has been damaged.

Cannot recover . entry, processing continued

[Chkdsk]

The "." entry (working directory) is defective and cannot be recovered.

Cannot recover ".." entry, Entry has a bad attribute (or link or size)

[Chkdsk]

The ".." entry (parent directory) is defective and cannot be recovered.

If you have specified the /f switch, chkdsk tries to correct the error automatically.

Cannot RECOVER a Network drive

[Recover]

You cannot recover files on drives that are redirected over the network.

Cannot setup expanded memory

[Fastopen]

This message indicates that the expanded memory (EMS) card in your system is not functioning properly.

Cannot specify default drive

[Sys]

If you don't leave enough space on the disk for Sys to install the system files and two unused entries in the root directory of the drive to which you want to transfer the operating system files, then Sys won't be able to specify the disk drive you want.

Cannot start COMMAND, exiting

[MS-DOS]

Either the value of Files in config.sys needs to be increased before restarting MS-DOS, the path to find the command processor was set incorrectly, or there was insufficient memory available to load the command processor.

Cannot SUBST a Network drive

[Subst]

You cannot substitute drives that are redirected over the network.

Cannot SYS to a Network drive

[Sys]

You cannot transfer the system files to drives that are redirected over the network.

For more information about the net print command, see the Microsoft Networks User's Guide.

Cannot use FASTOPEN for drive x:

[Fastopen]

Fastopen works only with local, hard disks and can work with a maximum of four disks at a time. You may be trying to use Fastopen over a network, with a floppy disk, or with more than four disks at one time, none of which is possible with Fastopen.

Cannot use PRINT - Use NET PRINT

[Print]

You must use the Net Print command to print files.

Cannot XCOPY to a reserved device

[Xcopy]

You must specify as your target a file or block device (not a character device) that is not NUL and, then, enter the Xcopy command again.

CHDIR .. failed, trying alternate method

[Chkdsk]

When checking the tree structure, Chkdsk was not able to return to a parent directory. It will try to return to that directory by starting over at the root and searching again.

CHKDSK not available on drive X

[Chkdsk]

This is an informational message indicating that you are trying to Chkdsk an alternate file system which cannot be found.

xxxxxxx code page drive cannot be initialized

[MS-DOS]

MS-DOS cannot start either the printer.sys or display.sys program. Check the device command line in your config.sys file. You probably included an illegal parameter. For more information, see Appendix C, Installable Device Drivers.

Code page not prepared**[Mode]**

You have selected a code page that has not yet been prepared for the system, or one that does not have the correct font to support the current video mode.

To prepare a code page for the system, use the mode prepare command. If you have installed the display.sys installable device driver, be sure the device command line in your config.sys file allows for additional subfonts. For more information, see Appendix C, Installable Device Drivers, and Appendix B, How to Configure Your System.

Code page xxx not prepared for all devices**[Chcp]**

You have selected a code page that is not currently supported by a device.

First, be sure your device supports code-page switching, and that it is currently on-line. If the device supports code-page switching, use the mode prepare command to prepare the device for the code page. Then retry the chcp command.

Code page xxx not prepared for system**[Chcp]**

Chcp is unable to select a code page for the system.

First, make sure that nlsfunc is installed. If you have not used the device command in your config.sys file to install device drivers, you may now retry the chcp command. If you are using installable device drivers with your system, you must use the mode prepare command to prepare the specific code page for each device on your system. Then retype the chcp command.

Code page operation not supported on this device**[Mode]**

You have specified a device and code page combination which MS-DOS does not recognize as valid.

Check to see that the device you specified exists and that you have listed a valid code page. Also check to see that that code page is supported on that device.

Code page requested xxx is not valid for given keyboard code**[Keyb]**

The keyboard code and code page specified are not compatible.

Retype the keyb command with a compatible keyboard code and code page.

Code page specified has not been designated

[Keyb]

You have typed the Keyb command with an option the system doesn't recognize. You must first prepare the associated code page for your console screen device.

Use the mode prepare command to prepare the associated code page for CON. Then retype the keyb command.

Code page specified has not been prepared

[Keyb]

You have typed the Keyb command with an option the system doesn't recognize. You must first prepare the associated code page for your console screen device.

Use the mode prepare command to prepare the associated code page for CON. Then retype the keyb command.

Code page specified is inconsistent with invoked code page

[Keyb]

This warning message lets you know that the Keyb option you've selected does not coincide with the code page for your console screen device (CON).

Use the mode select command if you also want to change the code page for CON.

Code page specified is inconsistent with selected code page

[Keyb]

This warning message lets you know that the Keyb option you've selected does not coincide with the code page for your console screen device (CON).

Use the mode select command if you also want to change the code page for CON.

Code page xxx

[Mode]

This is the code page currently being used by the device specified.

Code pages cannot be prepared

[Mode]

You have either specified a duplicate code page for this device or tried to prepare more than the total number of code pages supported for this device.

Check the device command line in your config.sys file to see how many prepared code pages are allowed for this device. Use the /status option of the mode command to find out which code pages are already prepared for this device. For more information, see Appendix B, How to Configure Your System, and Chapter 3, MS-DOS Commands.

Compare another diskette (Y/N)?

[Diskcomp]

*Diskcomp displays this message when it has completed its comparison of the disks.**Type Y (for Yes) if you want to compare more disks; or type N (for No) if you don't.***Compare error at OFFSET X**

[Comp]

*This is an informational message indicating that the files Comp is comparing contain different hexadecimal values at the displayed offset (also in hexadecimal) into the file.***Compare error on disk side s, track t**

[Diskcomp]

*Diskcomp found a difference on the disk in the specified drive, side s, track t.***Compare more files (Y/N)?**

[Comp]

*Comp displays this message when it has completed its comparison of the files.**Type Y (for Yes) if you want to compare more files; or type N (for No) if you don't.***Compare OK**

[Diskcomp]

*Diskcomp displays this message if the disks are identical.***Compare process ended**

[Diskcomp]

*Diskcomp displays this message if a fatal error occurred during the comparison.***Comparing t tracks n sectors per track, s side(s)**

[Diskcomp]

*This message confirms the format of the disks that you are comparing.***COM port does not exist**

[Mode]

You have specified an invalid COM port.

X Contains N non-contiguous blocks

[Chkdsk]

The disk contains fragmented files.

If you want to copy this disk, you should use the copy or xcopy command instead of the diskcopy command. The new copy will then store the new files sequentially.

Content of destination lost before copy

[Copy]

The source file that you specified in the Copy command was overwritten before the copy process completed.

Refer to the copy command for the proper syntax.

Convert directory to file (Y/N)?

[Chkdsk]

The directory is no longer usable because it contains too much invalid information.

Type Y (for Yes) if you want to convert it to a file and then use debug, or type N (for No) if you don't.

Convert lost chains to files (Y/N)?

[Chkdsk]

Chkdsk displays this message if it finds information on the disk that isn't allocated properly in the disk's File Allocation Table.

If you type Y (for Yes) in response to this prompt, chkdsk recovers the lost blocks it found when checking the disk. Chkdsk then creates a proper directory entry and a file for each lost chain with a filename of the form: filennnn.chk. If you type N (for No), chkdsk frees the lost blocks so that they can be reallocated and does not recover any data that was in those lost blocks.

Copy another diskette (Y/N)?

[Diskcopy]

The Diskcopy command has completed processing.

Type Y (for Yes) if you want to copy another disk, or type N (for No) if you don't.

Copying t tracks n Sectors/Track, s Sides

[Diskcopy]

Diskcopy displays this message during copying.

Copy process ended

[Diskcopy]

Diskcopy could not copy the entire disk.

Use the copy or xcopy command to copy specific files onto the disk.

Corrections will not be written to disk

[Chkdsk]

There are errors on the disk, but Chkdsk will not correct them because you did not specify the /f switch.

You must specify the chkdsk /f switch to correct disk errors.

Current code page settings

[Mode]

This informational message shows current and prepared code pages for the device specified and for the system.

Current date is mm-dd-yy

[Date]

The Date command displays this message.

Enter the correct date and press the ENTER key.

Current drive is no longer valid

[MS-DOS]

Your current drive is either a network drive or a disk drive with no disk in it.

You must either change to a valid drive or, if using a disk drive, make sure you have a disk in it and that the drive door is closed.

Current keyboard code: xx code page: yyy

Current CON code page: zzz

[Keyb]

This message displays the current keyboard code and its associated code page, and the current code page used by your console screen device (CON).

Current keyboard does not support this code page

[Keyb]

The code page selected is not compatible with the current keyboard code.

Check the code page you have selected. If it is correct, change the keyboard code with the keyb command.

Current time is hh:mm:ss.cc

[Time]

The Time command displays this message.

Enter the correct time and press the ENTER key.

Data error reading drive x:

[MS-DOS device error]

MS-DOS could not read the data from the disk properly. This is often due to a defective disk.

Try typing R (for Retry) several times, or type A (for Abort) to end the program. (It's a good idea to make a new copy of the disk, because if it's defective, you may lose information.)

Delete (Y/N)?

[Command]

You requested a deletion.

Type a Y (for Yes) to erase the file, or type an N (for No) to advance to the next file.

Delete current volume label (Y/N)?

[Label]

If a current volume label exists, Label displays this message in response to the prompt to enter the new volume label for this disk.

If you want to delete the volume label, type Y (for Yes); otherwise, type N (for No).

Device Error during Status

[Mode]

MS-DOS found an error with the specified device when it was checking the status of that device. The problem may be due to a device that does not support code pages, a device not properly prepared for code-page switching, a device which cannot support more code pages than those already prepared, or a device with a bad or irregular font file.

Check the device command line in your config.sys file. Make sure that the command syntax and limits for subfonts and additional code pages are all correct. Also check to see if your device supports code-page switching. Consult the hardware vendor if you are unsure.

Device Error during Prepare

[Mode]

MS-DOS found an error with the specified device when preparing that device for code-page switching. The problem may be due to a device that does not support code pages, a device not properly prepared for code-page switching, a device which cannot support more code pages than those already prepared, or a device with a bad or irregular font file.

Check the device command line in your config.sys file. Make sure that the command syntax and limits for subfonts and additional code pages are all correct. Also check to see if your device supports code-page switching. Consult the hardware vendor if you are unsure.

Device Error during Select**[Mode]**

MS-DOS found an error with the specified device. The problem may be due to a device that does not support code pages, a device not properly prepared for code-page switching, a device which cannot support more code pages than those already prepared, or a device with a bad or irregular font file.

Check the device command line in your config.sys file. Make sure that the command syntax and limits for subfonts and additional code pages are all correct. Also check to see if your device supports code-page switching. Consult the hardware vendor if you are unsure.

Device Error during write of font file to device**[Mode]**

MS-DOS found an error when it tried to write the font file to the specified device. The problem may be due to a device that does not support code pages, a device not properly prepared for code-page switching, a device which cannot support more code pages than those already prepared, or a device with a bad or irregular font file.

Check the device command line in your config.sys file. Make sure that the command syntax and limits for subfonts and additional code pages are all correct. Also check to see if your device supports code-page switching. Consult the hardware vendor if you are unsure.

Device or code page missing from font file**[Mode]**

MS-DOS did not find a definition of the indicated code page for this device in the font file.

Use the mode command to specify another code page for this device. Also check to see that the font file supports the code page you want to use. This error also may cause specified code pages to be undefined. Use the mode command to prepare and refresh lost code pages.

Device ddd not prepared**[Mode]**

No code page has been prepared for this device.

Directory is joined

[Chkdsk]

Chkdsk does not process directories that are joined.

Use the join /d command to unjoin the directories, and then run chkdsk again.

Directory is totally empty, no . or ..

[Chkdsk]

The specified directory does not contain references to working and parent directories.

Delete the specified directory and recreate it.

Directory not empty

[Join]

You can only join onto an empty directory.

Disk error reading (or writing) drive x:

[MS-DOS device error]

MS-DOS could not read the data from the disk properly. This is often due to a defective disk.

Try typing R (for Retry) several times, or type A (for Abort) to end the program. (It's a good idea to make a new copy of the disk, because if it's defective, you may lose information.)

Disk error reading (or writing) FAT X

[Chkdsk]

One of your File Allocation Tables has a defective sector in it. MS-DOS automatically uses the other FAT.

You should copy all your files onto another disk. To correct this error automatically, you simply specify the chkdsk /f switch.

Disk full. Edits lost

[Edlin]

Edlin was not able to save your file due to lack of disk space.

You should always make sure that there is enough room on the default disk to save your file before you use the Edlin E (end) command. You should also make sure that the default disk is not write-protected.

Disk full error writing to BACKUP Log File

[Backup]

The disk to which you were writing the log file is full.

To restart, press any key or CONTROL+BREAK.

Disk unsuitable for system disk

[Format]

The Format program detected a bad track on the disk where system files should reside.

You should use this disk to store data only.

Do not specify filename(s)

Command format: DISKCOMP d: d:[/1][/8]

[Diskcomp]

You specified an incorrect switch or gave a filename in addition to a drive name.

Do not specify filename(s)

Command format: DISKCOPY d: d:[/1]

[Diskcopy]

You specified an incorrect switch or gave a filename in addition to a drive name.

Do you see the leftmost 0? (Y/N)

[Mode]

Mode displays this message to help you align the test pattern on your screen.

Type Y (for Yes) if you can see the leftmost 0 in the test pattern, or type N (for No) if you want to shift the display to the right.

Do you see the rightmost 9? (Y/N)

[Mode]

Mode displays this message to help you align the test pattern on your screen.

Type Y (for Yes) if you can see the rightmost 9 in the test pattern, or type N (for No) if you want to shift the display to the left.

Do you wish to use the maximum size for a DOS partition and make the DOS partition active (Y/N).....[]

[Fdisk]

You are formatting your hard disk.

Type Y (for Yes) and press the ENTER key if you want to format your entire hard disk as the primary DOS partition. Otherwise, type N (for No) and press the ENTER key.

Does name specify a file name or directory name on the target (F = file D = directory)?

[Xcopy]

Xcopy displays this prompt if the target directory does not exist.

Type F if the name specifies a file, or D if the target specifies a directory that does not currently exist.

(.)(..) Does not exist

[Chkdsk]

This is an informational message from Chkdsk, indicating that either the "." or ".." directory entry is invalid.

DOS command line parameters supported

[Graftabl]

This is an informational message. Graftabl will display command line parameters.

Drive x already deleted

[Fdisk]

You tried to delete drive X, but it had already been deleted.

Drive deleted

[Fdisk]

You deleted a hard drive from the system.

Drive has been changed or deleted

[Fdisk]

You changed or deleted a hard drive on the system.

Drive letter must be specified

[Format]

You did not specify the drive letter for the drive that you want to format.

You must specify the name of the drive to format.

Drive types or diskette types not compatible

[Diskcomp][Diskcopy]

You must have the same size and type of disks to run these commands. For example, you cannot copy from a single-sided disk to a double-sided disk, or compare a high-density disk with a low-density disk.

You should use Fc if you want to compare the files on the disks. If you want to copy the disk, you can use copy or xcopy. You could also reformat the target disk so that it's the same type as the source disk, or use a disk of the same type.

Duplicate file name or file not found

[Rename]

You tried to rename a file to a filename that already exists, or the name you specified could not be found.

Duplicate parameters not allowed

[Graphics]

You cannot specify a parameter twice.

ECHO Is off (or on)

[MS-DOS]

This message tells you the current status of Echo.

End of Input file

[Edlin]

The entire file was read into memory. If the file was read in sections, this message indicates that the last section of the file is in memory.

Enter current Volume Label for drive x:

[Format]

Format asks you to enter the current volume label for verification before it formats the hard disk in the specified drive.

If you do not know what the volume label is, press CONTROL+C to abort this command, and give the vol command for the specified drive. Then give the format command again.

Enter new date:

[Date]

You must respond to this prompt when you start MS-DOS, or when you use the Date command.

Type the date in the format indicated by the prompt, or press the ENTER key to accept the current date.

Enter new time:

[Time]

You must respond to this prompt when you start MS-DOS.

Type the time in the format indicated by the prompt, or press the ENTER key to accept the current time.

Entry error

[Edlin]

The last command you typed contained a syntax error.

Retype the command with the correct syntax and press the ENTER key.

Entry has a bad attribute (or link or size)

[Chkdsk]

This message may be preceded by one or two periods that show which subdirectory is invalid.

If you have specified the /f switch, chkdsk tries to correct the error automatically.

EOF mark not found

[Comp]

While comparing non-text files, Comp could not find the end of the valid data in the last block of files being compared.

Error during read of Font file

[Mode]

MS-DOS found an error when it tried to read the font file for the code page specified.

Error in country command

[MS-DOS]

You used the incorrect syntax for the Country command in your config.sys file.

For the correct syntax of this configuration command, see Appendix B, How to Configure Your System.

Error in .EXE file

[MS-DOS]

The .exe file you have asked MS-DOS to load has an invalid internal format.

You cannot run this program. Check to make sure that you are using the correct version of MS-DOS.

Error loading operating system

[Fdisk]

You were unable to load your operating system from the hard disk due to a disk error.

If you restart the system and the error continues, restart it again using your MS-DOS disk and then the sys command to transfer a new copy of MS-DOS to your hard disk.

Error opening log file

[Backup]

MS-DOS cannot open the backup log file.

Check the drive and path specified with this command. Remember that the log file cannot be located on the target drive. If you did not specify a filename for the log, then the error occurred when MS-DOS tried to open and create backup.log on the source disk.

Error reading directory

[Format]

You could not read the directory due to bad sectors within the directory or FAT structure.

If the error occurred on the hard disk, try to format it again to make it usable; otherwise, throw it out.

Error reading fixed disk

[Fdisk]

After five tries, Fdisk still cannot read the startup record of the current hard disk.

Consult your system's manuals or call your vendor.

Error reading GRAPHICS profile

[Graphics]

MS-DOS cannot read the Graphics profile.

Make sure the disk drive door is closed.

Error reading/writing partition table

[Format]

Format could not read or write the partition table.

You should run fdisk on the disk and then try formatting it again.

Errors found, F parameter not specified

Corrections will not be written to disk

[Chkdsk]

Chkdsk found errors on the disk. If you have not specified the /f switch, Chkdsk continues printing messages but will not correct the errors.

You should run chkdsk with the /f switch if you want to correct the problems encountered by the chkdsk command.

Errors on list device indicate that it may be off-line.

Please check it.

[Print]

Your printer is not turned on.

Error writing directory (FAT or fixed disk)

[Format][Fdisk]

Either the File Allocation Table (FAT) or the directory has bad sectors.

If the error occurred on your hard disk, try to format it again to make it usable; otherwise, throw it out.

Error writing to device

[MS-DOS]

You tried to send too much data to a device, so MS-DOS was unable to write the data to that device.

EXEC failure

[MS-DOS]

MS-DOS either found an error when reading a command, or the Files command in the config.sys file is set too low.

Increase the value of the files command in the config.sys file, and restart MS-DOS.

Expanded Memory not available

[MS-DOS]

This is an informational message indicating that you do not have an expanded memory card installed in your system.

Extended DOS partition already exists

[Fdisk]

You cannot create another extended DOS partition.

Extended DOS partition created

[Fdisk]

You have created an extended DOS partition on your hard disk.

Extended DOS partition deleted

[Fdisk]

You have deleted an extended DOS partition from your hard disk.

Failure to access code page font file

[Mode]

MS-DOS cannot open the font file for the specified code page.

Check to see that you typed font file name and its pathname correctly. Also check the config.sys file to see that the device driver for this device has been properly installed. If the config.sys file is incorrect, correct it and restart MS-DOS before retyping the mode command.

Failure to access device: xxx

[Mode]

You are trying to specify a code page for a particular device, but MS-DOS cannot access the device listed.

Retype the command using an existing device. Make sure you are typing the device name correctly.

Fastopen already installed

[Fastopen]

Fastopen is already installed on the system.

Fastopen installed

[Fastopen]

This informational message acknowledges that you have just installed Fastopen.

FCB unavailable reading (or writing) drive x:

[MS-DOS device error]

An unusual error has occurred. This error usually requires an experienced programmer to fix it.

Type R (for Retry) or A (for Abort).

fc: cannot open filename - No such file or directory

[Fc]

One of the files that you specified doesn't exist.

Check the directory for the correct filename.

fc: filename longer than filename

[Fc]

After reaching the end of one of the files in a file comparison, the other file still has data left that was not yet compared.

fc: incompatible switches

[Fc]

You have specified switches that are not compatible. (For example, /b and /L.)

You should not combine binary and ASCII comparison switches.

fc: no differences encountered

[Fc]

The files are the same.

fc: out of memory

[Fc]

You do not have enough memory to perform the comparison.

File allocation table bad

[MS-DOS]

The disk may be defective.

Run chkdsk /f to check the disk.

File allocation table bad drive x:

[Chkdsk]

This message means that the disk was not formatted or was formatted improperly. It could also mean that an operating system other than MS-DOS is on the disk.

Run chkdsk /f to check the disk. If this message is displayed again, you must reformat the disk.

File filename canceled by operator

[Print]

MS-DOS displays this message when you specify the /t switch with the Print command.

File cannot be converted

[Exe2bin]

The input file is not in the correct format.

File cannot be copied onto itself

[Copy][Replace][Xcopy]

The source filename you specified is the same as the target filename.

File creation error

[MS-DOS][Edlin][Restore][Xcopy]

You tried to add a new filename or replace a file that already exists in the directory, or there was not enough space for the file. If the file already exists, it is a read-only file and cannot be replaced. This error message may also occur if the root directory is full, out of files, or if the filename is the same as a volume or directory, or a hidden (or system) file.

File is READ-ONLY

[Edlin]

The file is designated read-only, so you may not change it.

File name must be specified

[Edlin]

You did not specify a filename when you started Edlin.

You should type the edlin command followed by a filename.

File not found

[Chkdsk][Edlin][Fc][Find][Print][Recover][Rename][Xcopy]

MS-DOS could not find the file that you specified, or you tried to rename a file with a name already in the directory.

Check to see that you entered the filename correctly.

File not in PRINT queue

[Print]

The file that you specified was not in the print queue, so you cannot remove it from the queue.

Check to see that you entered the filename correctly.

Files are different sizes

[Comp]

This is an informational message indicating why Comp did not compare the files.

***** Files were backed up at time on date *****

[Restore]

This is an informational message only.

FIND: Access denied

[Find]

You cannot access the file.

Make sure that the disk is not write-protected, read-only, or locked.

FIND: File not found

[Find]

MS-DOS could not find the file that you specified.

Make sure you have typed the filename correctly.

FIND: Invalid number of parameters

[Find]

You specified either too many or too few options in the command line.

FIND: Invalid Parameter

[Find]

One of the switches you specified is wrong.

FIND: Read error in filename

[Find]

The Find command could not read the specified file.

FIND: Syntax error

[Find]

Check to make sure that you have typed the command correctly.

First cluster number is invalid, entry truncated

[Chkdsk]

The file directory entry contains an invalid pointer to the data area. If you specified the /f switch, the file is truncated to a zero-length file.

FIRST diskette bad or incompatible

[Diskcomp]

Diskcomp cannot recognize the format on the source disk. You should run chkdsk to help you identify the problem.

Fixed backup device X is full

[Backup]

This is an informational message indicating that you can no longer use that device for backing up files.

Fixups needed - base segment hex:

[Exe2bin]

The source (.exe) file contained information indicating that a load segment is required for the file.

You must specify the absolute segment address where the finished module is to be located.

Font File contents invalid

[Mode]

MS-DOS cannot use the contents of the font file specified.

Make sure you are typing the name of the font file correctly. Retype the command. If this message is displayed again, your font file may have been altered or corrupted. Recopy this file from the master MS-DOS disk. Type the command again. This error may also cause existing selected code pages to be undefined. Use the mode command to prepare these code pages again, and to refresh them.

For cannot be nested

[MS-DOS]

You cannot nest For commands in a batch file.

Format another (Y/N)?

[Format]

Format displays this message when it has finished formatting a disk. Type Y (for Yes) if you want to format another disk, or type N (for No) if you don't. If you accidentally type Y, you can abort the format process by typing CONTROL+C in response to the message Strike any key.

Format complete

[Format]

Format displays this message when it has finished formatting the disk in the specified drive.

Format failure

[Format]

MS-DOS could not format the disk. This message is usually displayed with an explanation as to why the command failed.

Format not supported on drive x:

[Format]

You cannot use Format to format this drive.

You may have specified device parameters that your computer cannot support.

Formatting while copying

[Diskcopy]

Diskcopy displays this message if the target disk has never been formatted.

Function not supported on this computer

[Mode]

Your machine cannot do the requested function due to a lack of the appropriate adapter or device.

General failure reading (or writing) drive x:

[MS-DOS device error]

An unusual error has occurred. This error usually requires an experienced programmer to fix it.

Type R (for Retry) or A (for Abort).

Graphics characters already loaded

[Graftabl]

The Graftabl command displays this message if you have already loaded the table of graphics characters into memory.

Graphics characters loaded

[Graftabl]

The Graftabl command displays this message after it loads the table of graphics characters into memory.

Hardware code pages:**Prepared code pages:**

[Mode]

This message lists the current code pages prepared for the device specified.

Has invalid cluster, file truncated

[Chkdsk]

The file directory entry contains an invalid pointer to the data area. If you specified the /f switch, the file is truncated to a zero-length file.

Illegal device name

[Mode]

Your computer does not recognize this device name.

Incompatible system size

[Sys]

The system files occupy more space on the source disk than is available on the target disk.

You cannot use the sys command to transfer the system files to this disk.

Incorrect APPEND Version

[Append]

You are not using the MS-DOS 4.0 Append command. You are using another incompatible version.

Incorrect DOS Version

[Append][Assign][Attrib][Backup][Chkdsk][Command]
[Comp][Diskcomp][Diskcopy][Exe2bin][Edlin][Fc][Fdisk][Find]
[Format][Graftabl][Graphics][Join][Keybxx][Label][Mem][Mode]
[More][Nlfunc][Print][Recover][Replace][Restore]
[Select][Share][Sort][Subst][Sys][Tree][Xcopy]

Some MS-DOS utilities will not run on older versions of the operating system, and many are written to run only on the exact version of MS-DOS that they were created for.

You must use the correct version of MS-DOS to run this command.

Incorrect DOS Version, use DOS 2.00 or later

[Link]

Some MS-DOS utilities will run only on MS-DOS version 2.00 or later versions.

Incorrect number of parameters

[Join][Subst]

You specified either too many or too few options in the command line.

Incorrect parameter

[Assign][Share]

One of the options you specified is wrong.

Infinite retry on parallel printer timeout

[Mode]

Your printer is probably off-line or not ready.

If the printer appears to be ready, you may have to press the CONTROL+ALT+DELETE keys to reset the computer.

Infinite retry not supported on network printer

[Mode]

Your system cannot sense printer errors through the network interface.

You should either use a printer with a direct connection or not specify retry.

Insert backup diskette n into drive x:

[Backup][Restore]

This message prompts you for the nth backup disk.

Put the next disk into the specified drive. Be sure to label each backup disk in the appropriate order for use when restoring the files.

Insert backup source diskette in drive X

[Backup]

This message prompts you for the backup source disk.

Insert disk with command.com in drive X

[Command]

This message prompts you for the disk with the command.com file on it.

Insert diskette for drive x: and strike any key when ready

[MS-DOS]

This message appears when MS-DOS is copying and formatting.

You should insert a disk in the appropriate drive and press any character or number key to begin processing.

Insert diskette with batch file and press any key when ready

[MS-DOS]

The disk containing your batch file is not in the drive you originally specified.

Reinsert the disk that contains the batch file in the appropriate drive.

Insert DOS diskette in drive x: and strike ENTER when ready

[Format]

You typed the Format /s command, but the disk in the default drive does not contain MS-DOS system files.

Insert a disk with the files io.sys and msdos.sys in the drive specified and press any key.

Insert FIRST diskette into drive x:

[Diskcomp]

This message prompts you for the first disk that you want to compare.

Insert last backup diskette in drive x: Strike any key when ready

[Backup]

This message prompts you for the final backup disk.

After you have put the final backup disk into the drive specified, press any alphanumeric key to continue the backup process.

Insert new diskette for drive x:

[Format]

This message prompts you for the disk you want to format.

Insert restore target diskette into drive x:

[Restore]

Restore displays this prompt if you are restoring files to a floppy.

Put the target disk into the specified drive.

Insert SECOND diskette in drive x:

[Diskcomp]

This message prompts you for the disk that you want to compare with the first disk.

Insert source disk

[Backup]

This message prompts you to put the source disk into the drive.

Insert SOURCE diskette into drive x:

[Diskcopy]

This message prompts you to put the disk to be copied into the specified drive.

Insert system diskette in drive x: and strike any key when ready

[Sys]

Sys needs a disk from which to read the io.sys and msdos.sys files.

Insert a system disk into the specified drive and press any character or number key to start the system copy process.

Insert TARGET diskette into drive x:

[Diskcopy]

Diskcopy displays this message to prompt you to place the target disk into the specified drive. If your computer has one floppy drive, this message prompts you to put the proper disk into the drive.

Insufficient disk space

[MS-DOS][Replace][Sort][Xcopy]

The disk is full and does not contain enough room to perform the specified operation.

Insufficient memory

[Backup][Chkdsk][Diskcomp][Diskcopy][Edlin][Replace]

[Restore][Sort][Xcopy]

There is not enough memory in your computer to perform the specified operation.

Before retrying this operation, you must free memory by deleting files. In Edlin, you may be able to free memory by typing a W (write) command followed by an A (append) command.

Insufficient room in root directory. Erase files in root and repeat CHKDSK

[Chkdsk]

Chkdsk always recovers lost files into the root directory. In this case, your root directory is full.

Delete some files in your root directory, or move them to another directory to make room for the lost files.

Intermediate file error during pipe

[MS-DOS]

The pipe operation uses temporary files on the disk that are deleted automatically once the piping process is complete. An error has occurred in one of these files.

Make sure that there is enough room on the disk for the temporary file and that the disk is not write-protected, and then try the command again.

Internal error

[Fc][Mode][Share]

This message indicates an error in the utility.

Internal stack overflow

System Halted

[MS-DOS]

The system tried to use more stacks than were available. This caused a series of hardware interrupts and halted the system.

Restart MS-DOS. Then edit your config.sys file and allocate more stack resources. For more information, see Appendix B, How to Configure Your System.

Invalid argument

[Backup][Fc][Restore]

You have specified an invalid argument.

Refer to Chapter 3, MS-DOS Commands, for the correct syntax of the command, and try again.

Invalid baud rate specified

[Mode]

You have specified an incorrect baud rate. Valid choices are 110, 150, 300, 600, 1200, 2400, 4800, and 9600.

You must specify at least the first two digits of the baud rate.

Invalid characters in volume label

[Format][Label]

The volume label should only contain up to 11 alphanumeric characters.

Invalid code page specified

[Chcp]

You selected an invalid code page number.

Retype the command with the correct code page.

Invalid combination of parameters

[Append][Assign][Attrib][Backup][Chkdsk][Command]
[Comp][Diskcomp][Diskcopy][Exe2bin][Edlin][Fc][Fdisk][Find]
[Format][Graftab][Graphics][Join][Keybxx][Label][Mem][Mode]
[More][Nlfunc][Print][Recover][Replace][Restore]
[Select][Share][Sort][Subst][Sys][Tree][Xcopy]

You must enter the command parameters in the correct order.

Invalid COMMAND.COM

Insert COMMAND.COM disk in default drive and strike any key when ready

[MS-DOS]

The program you have just run used up almost all of memory. MS-DOS must now reload the command.com file from disk. However, either MS-DOS cannot find command.com on the disk, or the copy it has found is the incorrect version.

Insert a disk that contains a copy of command.com into the default drive (it must be the same version with which you started MS-DOS).

Invalid country code or code page

[MS-DOS]

MS-DOS found an invalid country code or code page number in your config.sys file.

Correct the country command line in your config.sys file.

Invalid current directory

[Chkdsk]

Your disk has an invalid directory on it.

You may be able to recover some of the files on this disk by copying them with the copy command. Otherwise, you must replace the disk.

Invalid date

[Date][Xcopy]

You specified an invalid date in response to the date prompt.

Enter a valid date. Refer to Chapter 3, MS-DOS Commands, for the proper syntax of the date command.

Invalid Date/Time

[Backup]

You specified an invalid date with one of the Backup command switches.

Refer to Chapter 3, MS-DOS Commands, for the proper syntax of the backup command, then try again.

Invalid device

[MS-DOS]

The device specified was not AUX, CON, NUL, or PRN.

Invalid device parameters from device driver

[Format]

Format displays this message when the number of hidden sectors is not evenly divisible by the number of sectors per track (that is, the partition does not start on a track boundary). This might happen if you tried to format a hard disk that previously had been formatted with MS-DOS 2.x without first running Fdisk, or if you have set the device driver parameters incorrectly.

Check the config.sys file for incorrect device or drivparm commands.

Invalid directory

[MS-DOS]

The directory you specified either does not exist or is invalid.

Check to see that you entered the directory name correctly.

Invalid disk change reading (or writing) drive x:

[MS-DOS device error]

You changed the disk in a drive when you weren't supposed to.

Put the disk back in the drive and type R (for Retry).

Invalid drive in search path

[MS-DOS]

The drive does not exist.

Invalid drive or filename

[Edlin][Recover]

You did not type a valid drive name or filename. Enter a valid drive name or filename.

Invalid drive specification

[Backup][Chkdsk][Diskcomp][Diskcopy][Format][Label]

[Print][Replace][Restore][Sys][Tree][Xcopy]

The drive is incorrect or does not exist.

Enter a valid drive name.

Invalid entry, please enter x

[Fdisk]

You must enter a value that is within the acceptable range.

Invalid entry, please press Enter

[Fdisk]

The last character you can enter is the percent sign.

If you want to enter additional characters, press ENTER or BACK-SPACE to change your entry and then press ENTER again.

Invalid environment size specified

[Command]

You gave an invalid number of bytes with the /e switch.

You must specify a number between 160 and 32,768 (bytes).

Invalid extent entry

[Fastopen]

You must enter a value for m that is within the appropriate range of from 1 through 999.

Invalid file/directory entry

[Fastopen]

You must enter a value of n that is within the appropriate range of from 10 through 999. If you do not provide a value, the default is 34.

Invalid keyboard code specified

[Keyb]

You selected an invalid keyboard code with the Keyb command.

Retype the command with the correct keyboard code.

Invalid number of parameters

[Attrib][Backup][Fc][Find][Recover][Restore][Xcopy]

Either you did not specify an option or string, or you specified the wrong number of options in the command line.

Invalid parameter (or parameter combination)

[Chkdsk][Fastopen][Append][Replace]

One of the switches you specified is wrong or does not exist.

Refer to Chapter 3, MS-DOS Commands, to make sure you are using the correct switches.

Invalid parameter(s)

[Backup][Chkdsk][Diskcomp][Diskcopy][Edlin][Find]

[Format][Join][Mode][Print][Replace][Restore][Sort]

[Subst][Sys][Tree][Xcopy]

One of the switches you specified is wrong or does not exist.

Refer to Chapter 3, MS-DOS Commands, to make sure you are using the correct switches.

Invalid partition table

[Fdisk]

MS-DOS found invalid information in the hard disk's partition information.

Use fdisk to examine and correct it.

Invalid path

[Append][MS-DOS]

The path is not correct.

Make sure the path is less than 63 characters and that the filename characters and format are valid.

Invalid path, not directory, or directory not empty

[MS-DOS]

You are unable to remove the directory requested for one of the specified reasons.

Invalid path (or file not found)

[Attrib][Backup][Copy][Restore][Tree][Xcopy]

You have entered a pathname or filename that does not exist.

Enter a valid pathname or filename with the command.

Invalid path or parameter

[Append]

You specified a file or directory that does not exist.

Enter a valid pathname or filename with the append command.

Invalid profile statement on line X

[Graphics]

You used an invalid word in the profile statement on line x.

Enter a valid word on the specified line.

Invalid STACK parameter

[MS-DOS]

The syntax of the Stack command in your config.sys file includes an invalid parameter.

See Appendix B, How to Configure Your System, for the correct syntax of that configuration command.

Invalid sub-directory entry

[Chkdsk]

The subdirectory that you specified either does not exist or is invalid.

Check whether you typed the subdirectory name correctly.

Invalid switch type

[Fastopen]

You specified a switch that is either not valid, in the wrong order, or duplicated.

Check the switches specified and correct them.

Invalid syntax**[MS-DOS]**

You used the wrong syntax when typing a command.

See Chapter 3, MS-DOS Commands, for the correct syntax of an MS-DOS command.

Invalid syntax on DISPLAY.SYS code page driver**[MS-DOS]**

You used the wrong syntax when you typed the Device command in your config.sys file to load display.sys.

See Appendix B, How to Configure Your System, for the correct syntax of device. Also see Appendix C, Installable Device Drivers, for information about the display.sys installable device driver.

Invalid syntax on PRINTER.SYS code page driver**[MS-DOS]**

You used the wrong syntax when you typed the Device command in your config.sys file to load printer.sys.

See Appendix B, How to Configure Your System, for the correct syntax of device. Also see Appendix C, Installable Device Drivers, for information about the printer.sys installable device driver.

Invalid syntax on PRINTER.SYS code page switching device drivers**[Mode]**

You used the wrong syntax when you typed the Device command in your config.sys file to load printer.sys.

See Appendix B, How to Configure Your System, for the correct syntax of device. Also see Appendix C, Installable Device Drivers, for information about the printer.sys installable device driver.

Invalid time**[Time]**

You specified an invalid time.

Refer to Chapter 3, MS-DOS Commands, for the correct syntax, and try the command again.

Invalid Volume ID**[Format]**

Format displays this message if you enter a volume label that doesn't match the label on the hard disk you want to format. It then quits the format process.

Use the vol command to find out what the volume label for the hard disk is, then try the command again.

Is cross linked on allocation unit X

[Chkdsk]

Chkdsk found two files using the same allocation unit.

You can use the /f parameter, which may truncate the file.

x Is not a choice, Please enter y-z

[Fdisk]

You tried to select an invalid option x.

Select a valid option from the range shown (y-z).

KEYB has not been installed

[MS-DOS]

No alternate keyboard code has been installed for your system.

If you want to use a keyboard code other than the default U.S.

(QWERTY) keyboard, use the keyb command to install it.

Label not found

[MS-DOS]

Your batch file contains a Goto command to a nonexistent label.

Last backup diskette not inserted Insert last backup diskette in drive x:

Strike any key when ready

[Backup]

This message prompts you for the final backup disk.

After you have put the final backup disk into the drive specified, press any alphanumeric key to continue the backup process.

***** Last file not backed up *****

[Backup]

Backup could not back up the last file on the disk. This message may occur if there is no more room on the target disk. It may also occur if there was an error in the source file or on the target disk.

You may have to back up this file separately to another disk.

Line too long

[Edlin]

During an Edlin R (replace) command, the string given as the replacement caused the line to expand beyond the limit of 253 characters.

You should divide the long line into two lines and retry the R command.

List output is not assigned to a device

[Print]

When you first type the Print command, MS-DOS asks you what device you want to specify as a printer. This message appears if Print is set up for a device that does not exist.

Lock violation reading (or writing) drive x:

[MS-DOS device error]

A program tried to access part of a file that another program was using.

Type A (for Abort), or wait awhile and type R (for Retry).

x lost allocation units found in y chains

Convert lost chains to files (Y/N)?

[Chkdsk]

Chkdsk displays this message if it finds information on the disk that isn't allocated properly in the disk's File Allocation Table.

If you type Y (for Yes) in response to this prompt, chkdsk recovers the lost blocks it found when checking the disk. Chkdsk then creates a proper directory entry and a file for each lost chain with the filename of the form: filennnn.chk. If you did not specify the /f switch, chkdsk displays: x bytes would be freed. If you type N (for No), chkdsk frees the lost blocks so that they can be reallocated and does not recover any data that was in those lost blocks. If you did not specify the /f switch, chkdsk does nothing.

Logging to file x

[Backup]

The Backup command is writing a backup log to the file specified.

Logical DOS drive created, drive letters changed or added

[Fdisk]

You have created or revised one or more logical drives.

LPTx: not rerouted

[Mode]

Mode could not reroute the parallel printer port.

Check to see whether you have specified the proper options.

LPTx: rerouted to COM x:

[Mode]

Output on the parallel printer port will now be sent to this asynchronous communications port.

LPTx: set for 80

[Mode]

The parallel printer port has been set for 80 columns.

LPTx: set for 132

[Mode]

The parallel printer port has been set for 132 columns.

Maximum number of logical DOS drives installed

[Fdisk]

You have installed the maximum number of logical DOS drives allowed by MS-DOS.

You may not create any more logical DOS drives.

Memory allocation error. Cannot load MS-DOS, system halted

[MS-DOS]

Restart MS-DOS.

If this error persists, make a new copy of the MS-DOS disk from your backup copy of the system disk.

Microsoft SMARTDrive RAM Cache version x.xx

Cache size: yyyyK in Expanded Memory

[Smartdrv.sys]

This is an informational message.

Mode fff code page function complete

[Mode]

This message is informational only.

—More—

[More]

Press the SPACEBAR to view more of the file or directory.

MORE: Incorrect DOS version

[More]

The More command does not run on MS-DOS versions before 2.0.

MS-DOS Version 4.00

Copyright International Business Machines Corp 1981, 1988

Copyright Microsoft Corp 1981-1986

[Command]

This message appears on most MS-DOS utility and command banners.

Must enter both /T and /N parameters

[Format]

You must enter both parameters together.

Must specify COM1, COM2, COM3 or COM4

[Mode]

You must specify a serial port.

Must specify destination line number

[Edlin]

You did not specify the destination line number for an Edlin C (copy) or M (move) command.

Retype the command with a destination line number.

Must specify ON or OFF

[MS-DOS]

The command requires either an ON or an OFF argument.

Name of list device [PRN]:

[Print]

This prompt appears the first time that Print is run and the /d switch is not specified.

You can specify the name of any valid device, which then becomes the print output device. If you press the ENTER key, MS-DOS uses the default list device PRN.

New file

[Edlin]

Edlin prints this message if it does not find a file with the name you specified.

If you are creating a new file, ignore this message. If you do not intend to create a new file, check to see whether you have correctly typed the name of the file that you wish to edit.

NLSFUNC already installed

[Nlsfunc]

Nlsfunc stays resident in memory once it is initialized. You have already loaded it into memory.

No Append

[Append]

No paths have been appended.

If you would like to append a path for data files, use the append command.

No appended directories

[Append]

You did not specify a path with the Append command.

No code page has been selected

[Chcp]

No code pages have been selected for the system.

If you would like to select a code page, use the chcp command.

No drive specifled

[Driver.sys]

You must specify the physical drive number in the Device command line in your config.sys file.

For further information, see the Driver.sys section in Appendix C, Installable Device Drivers.

No Extended DOS Partition to delete

[Fdisk]

You chose the wrong option for the current hard disk.

To review information on your hard disk, choose Display Partition Data from the fdisk Options menu.

No files added (or replaced)

[Replace]

The Replace command did not add or replace any files.

No files found filename

[Replace]

Replace could not find matching source or target files.

No fixed disks present

[Fdisk]

You must determine the cause of the problem and act appropriately.

**No free file handles. Cannot start
COMMAND.COM, exiting**

[MS-DOS]

Restart MS-DOS.

If this message recurs, increase the files command value in the config.sys file.

No Logical DOS Drive(s) to delete

[Fdisk]

This is an informational message indicating that there are no logical DOS drivers in this partition.

Press the ESC key to return to the previous menu.

No logical drives defined

[Fdisk]

There are no logical drives defined for your system.

No paper error writing device dev

[MS-DOS device error]

The printer is either out of paper or not turned on.

No partitions to delete

[Fdisk]

You chose the wrong option for this hard disk.

Choose another option from the fdisk Options menu.

No partitions to make active

[Fdisk]

You chose the wrong option for this hard disk.

From the fdisk Options menu, choose Create DOS partition or logical DOS drive to create a partition, and then Set active partition to make it the active one.

No path

[Path]

You typed Path and pressed the ENTER key to find out what your search path is, but you didn't set a command search path.

No primary DOS partition to delete

[Fdisk]

You have selected the Fdisk option to delete your primary DOS partition, but that partition does not exist.

No room for system on destination disk

[Sys]

There is not enough room for the system files on the target disk.

Delete some files to make room for the system files or use another disk. You may need to reformat the disk to put the system on it.

No room in directory for file

[Edlin]

You tried to create or save a file to the root directory, but it is either full, or you specified an invalid disk drive or filename.

Check the command line that you used to start Edlin for an invalid filename or disk drive entry. If your command contains no invalid entries, you should run the chkdisk program for the specified disk drive. If the status report shows that the disk directory is full, and if there is still enough memory left on the disk, you may be able to create the file in a subdirectory. (This is because subdirectories are not limited in size as is the root directory.) Otherwise, remove the disk and replace it with another formatted disk.

No source drive specified

[Backup]

You must specify a source drive.

No space left on device

[Backup][Fc][Restore]

You cannot back up or restore any more files, and you cannot send any more output from a file comparison to your disk because the target disk is now full.

You should probably delete some of the files on the disk to make more room.

No space to create a DOS partition

[Fdisk]

You do not have enough space on your current hard disk to create a DOS partition.

Either remove or reduce the size of the existing partition, and then run fdisk again to create the DOS partition(s).

No space to create logical drive

[Fdisk]

You are trying to create a logical drive, but there is no space available to do so.

No sub-directories exist

[Tree]

You have specified the /s switch, but the directory does not contain subdirectories.

No such file or directory

[Backup][Fc][Restore]

One or more of the files or directories that you specified does not exist.

No system on default drive

[Sys]

You must use a disk with the system files on it (the DOS disk) and try again.

No target drive specified

[Backup]

You must specify a target drive for this command.

Non-DOS disk error reading (or writing) drive x:

[MS-DOS device error]

MS-DOS does not recognize the disk format because the disk is missing information or contains another operating system.

Try running the chkdisk command to correct the problem. (See Chapter 3, MS-DOS Commands, for information about chkdisk.) If running chkdisk does not solve the problem, you should reformat the disk by using the format command—even though this will destroy all the files on the disk.

Non-system disk or disk error

Replace and strike any key when ready

[Format][Sys]

Replace the disk with the proper disk and press any alphanumeric key to continue.

***** Not able to back up (or restore) file *****

[Backup]

This message may occur if there was an error in the source file or on the target disk.

Use the chkdisk command on the source disk to see if you can determine the problem.

Not enough memory

[Join][Share][Subst]

There is not enough memory for MS-DOS to run the command.

Not enough room to merge the entire file

[Edlin]

There was not enough room in memory to hold the file during an Edlin T (transfer) command.

You must free some memory by writing some files to a disk or by deleting some files before transferring this file.

Not found

[Edlin]

You specified an Edlin S (search) or R (replace) command that was unable to find a further occurrence of the specified search or replace string.

Not ready error reading (or writing) drive x:

[MS-DOS device error]

The device (usually a drive or printer) specified in the error message is not ready to accept or transmit data.

This often happens when the disk drive door is open. If this is the problem, close the door and type R (for Retry), or check to see if the printer is on and ready to print.

O.K.?

[Edlin]

This prompt occurs during Edlin S (search) or R (replace) command processing.

If you press any key except Y (for Yes) or the ENTER key, the search or replace process continues.

One or more CON code pages invalid for given keyboard code

[Keyb]

Keyb examined all prepared code pages, and has found that at least one code page is incompatible for your screen console device (CON). This is only a warning to let you know that your keyboard and screen console device are working from different code pages.

Only non-bootable partitions exist

[Fdisk]

None of the partitions left can boot MS-DOS.

Only partitions on drive 1 can be made active

[Fdisk]

You are trying to create an active partition on a hard disk other than that found on the first hard disk drive. This is not allowed.

Out of environment space

[Command][MS-DOS]

There is not enough room in the program environment to accept more data.

To increase the size of the existing environment, use the /e switch with the command command or remove some of the existing environment variables by using the set command.

Page Frame starts at

[Xma2ems.sys]

This is an informational message showing the count of memory during testing.

Pages available for expanded memory

[Xma2ems.sys]

This is an informational message showing the number of 16K pages available for EMS mapping.

Pages mapped to conventional memory....

[Xma2ems.sys]

This is an informational message.

Parameter syntax or value error

[Xma2ems.sys]

This is an informational message showing that the syntax of the Device=Xma2ems.sys command in the config.sys file is incorrect.

Parameters not compatible

[Format][Replace]

You have specified switches that cannot be used together.

Parameters not compatible with fixed disk

[Format]

You have used a switch that is not compatible with the specified drive.

Parameters not supported

[MS-DOS][Format]

You have specified parameters that MS-DOS does not support.

Parameters not supported by Drive

[Format]

Format displays this message when the device driver for this drive does not support Generic IOCTL function requests.

**Partition selected (x) is not bootable,
active partition not changed**

[Fdisk]

You are trying to change active partitions, but MS-DOS cannot be booted from the partition selected.

Path(name) too long

[Print][Replace][Xcopy]

The pathname you specified was too long.

You may have to change directories to use this command with files in deep subdirectories.

Path not found

[Chkdsk][Replace][Subst][Xcopy]

You specified an invalid pathname.

X percent of disk formatted

[Format]

This is an informational message indicating the percentage of the disk already formatted.

Press any key to begin adding (replacing) file(s)

[Replace]

When you specify the /w switch, Replace displays this message to prompt you to start replacing files.

Press any key to begin copying file(s)

[Xcopy]

This message provides a pause to allow you to change disks and then press a key before copying begins.

Press any key to begin formatting x:

[Format]

This prompt is issued before you format a disk.

Press any key to begin the format process. Or, if you wish to end this command, press CONTROL+C.

Press any key to begin recovery of the n file(s) on drive x:

[Recover]

This prompt is issued before you recover a disk or file.

Press any key to begin the recovery. Recovered files are named filennnn.rec. If you wish to end this command, press CONTROL+C.

Press any key to continue

[MS-DOS]

When you enter an option or a command, MS-DOS prompts you to press a key to continue processing.

Press any key when ready . . .

[Diskcomp][Diskcopy]

This prompt gives you time to insert the appropriate disks before copying them.

When you have inserted the disks into the appropriate drives, press any key to begin the diskcopy process. Or, if you wish to end this command, press CONTROL+C.

Previous Code Page: x

[Graftabl]

This status message can indicate either that Graftabl was or was not used already to load a code character table and whether or not it has been altered.

Previously prepared code page replaced

[Mode]

This command changed the selected code page for a specific device by using another prepared code page.

Primary DOS partition already exists

[Fdisk]

You are trying to create a primary DOS partition, but one already exists.

If there is space available on your hard disk, try to create an extended DOS partition instead.

Primary DOS partition created

[Fdisk]

You have successfully created a primary DOS partition on your disk.

Primary DOS partition deleted

[Fdisk]

You have deleted the primary DOS partition from your disk.

Printbox ID not in GRAPHICS profile

[Graphics]

You must specify a printbox size (ID) that matches the first operand of a Printbox command in the Graphics profile.

Printer error

[Mode]

The printer is off or is not ready to print.

Printer lines per inch set

[Mode]

Mode has set the number of lines per inch for the printer.

PRINT queue is empty

[Print]

There are no files waiting to be printed.

PRINT queue is full

[Print]

There is only room for 10 files in the list of files waiting to be printed.

You can make room for more by using the print /q switch.

The limit is 32 files.

Printer type not in GRAPHICS profile

[Graphics]

You can only specify printer types that are defined in the Graphics profile.

See the graphics command in Chapter 3, MS-DOS Commands, for a list of the supported printer types.

Probable non-DOS disk Continue (Y/N)?

[Chkdsk]

The disk you are using is not recognized by this version of MS-DOS. The disk was either created by another system with a format that is not supported on this version of MS-DOS, or it is not an MS-DOS disk.

Do not continue processing if chkdsk returns this message for a floppy disk. If this message returns for a hard disk, the information describing the characteristics of the disk to MS-DOS has been destroyed. In this case, you may continue chkdsk processing by typing Y (for Yes). This error may mean that the File Allocation Table (FAT) is bad and that the disk is unusable.

Processing cannot continue

[Chkdsk]

There is not enough memory in your machine to run Chkdsk for this disk.

You must obtain more memory to run chkdsk.

Profile statement out of sequence on line X

[Graphics]

This message indicates where you need to correct the sequence of the profile statement.

Program too big to fit in memory

[MS-DOS]

You need more memory to run your application. It is possible that some programs you have run are still using some memory.

You may try to restart MS-DOS; however, if you still receive this message, you still need more memory.

RAMDrive: Computer must be PC-AT or PC-AT compatible

[Ramdrive.sys]

The /E switch was specified on a computer other than an IBM PC-AT (or 100% compatible), AT&T 6300 PLUS, or a 386 with extended memory. Ramdrive.sys was not installed.

Remove the /E parameter. Try installing RAMDrive in the system memory.

RAMDrive: Expanded Memory Manager not present

[Ramdrive.sys]

You included the /A switch in the RAMDrive command line. However, RAMDrive could not find the expanded memory manager because your config.sys file does not contain the appropriate information. Ramdrive.sys was not installed.

Consult your expanded memory card documentation for the correct installation instructions.

RAMDrive: Expanded Memory Status shows error

[Ramdrive.sys]

DOS detected an error in your expanded memory adapter while trying to install RAMDrive. Ramdrive.sys was not installed.

Consult your expanded memory adapter documentation on memory errors.

RAMDrive: Incorrect DOS version

[Ramdrive.sys]

RAMDrive requires DOS 2.x, DOS 3.x, or DOS 4.0. Windows 2.0 requires DOS 3.0 or higher. Ramdrive.sys was not installed.

Upgrade to DOS 3.0 or higher.

RAMDrive: Insufficient memory

[Ramdrive.sys]

Your computer has some memory available, but not enough to set up a RAMDrive. Ramdrive.sys was not installed.

Free some extended memory or obtain more memory.

RAMDrive: Invalid parameter

[Ramdrive.sys]

The parameters you specified in your config.sys entry for ramdrive.sys are not correct. Ramdrive.sys was not installed.

Check to see if you specified too many parameters, if one of your numeric parameters is not valid, if you specified conflicting switches (i.e., only one of either /E or /U may be specified), or if you specified too many switches. Change the ramdrive.sys command line in your config.sys file to conform to the usage described.

RAMDrive: I/O error accessing drive memory

[Ramdrive.sys]

During the set up of the RAMDrive, an error was detected in the memory being accessed for RAMDrive. Ramdrive.sys was not installed.

Run the memory test for the memory on which you were attempting to install a RAMDrive.

RAMDrive: No extended memory available

[Ramdrive.sys]

Your computer has no memory available for RAMDrives. Ramdrive.sys was not installed.

Free some extended memory or obtain more memory.

RATE and DELAY must be specified together

RATE=X

DELAY=X

[MODE]

You must specify both the typematic rate and the delay value when you change the setting of either.

Read error, COUNTRY.SYS

[MS-DOS]

MS-DOS cannot read the country.sys file.

Retry the command. If you get the same message, the country.sys file is probably corrupted. Restore the file from backup.

Read error in filename

[Edlin][Find]

MS-DOS could not read the entire file.

Read error, KEYBOARD.SYS

[MS-DOS]

MS-DOS cannot read the keyboard.sys file.

Retry the command. If you get the same message, the keyboard.sys file is probably corrupted. Restore the file from backup.

Read fault error reading drive x:

[MS-DOS]

MS-DOS is unable to read data from the device (usually a disk drive).

Check to see that the disk is properly inserted in the drive, then type R (for Retry).

Reading source file(s)...

[Xcopy]

Xcopy is now reading the source files that you specified.

Reinsert diskette for drive x:

[Format]

Reinsert the disk being formatted in the indicated drive.

Replace filename? (Y/N)

[Replace]

Replace displays this prompt if you specify the /w switch.

Type Y (for Yes) if you want to replace the existing file, or type N (for No) if you do not want to replace the file.

Replace the file (Y/N)?

[Restore]

The file that you want to restore from backup already exists on your target disk.

Type Y (for Yes) and press the ENTER key to overwrite the file.

Type N (for No) and press the ENTER key if you don't want to replace the file on your target disk with the file from the backup disk.

Replacing filename

[Replace]

Replace displays this prompt to let you know that it is replacing this file that exists on your disk.

Requested logical drive size exceeds the maximum available space

[Fdisk]

You are trying to create a logical drive that is larger than the space available.

Requested partition size exceeds the maximum available space

[Fdisk]

You are trying to create a partition on your hard drive that is larger than the space available.

Requested Screen Shift out of range

[Mode]

You cannot shift the display any farther.

Required font not loaded

[Mode]

Display.sys doesn't have the font size to do what you asked. Change the number of subfonts (m) to the maximum in the device=display.sys command line in your config.sys file, and then restart DOS.

Required profile statement missing before line X

[Graphics]

You left out a profile statement before line x and need to add it.

Resident part of PRINT Installed

[Print]

This is the first message that MS-DOS displays when you issue the Print command. It means that to process the Print command with other processes, available memory has been reduced by several thousand bytes.

Resident portion of MODE loaded

[Mode]

Part of the Mode program is now resident in memory, and available memory has been reduced by several thousand bytes.

Resident portion of NLSFUNC loaded

[Nlsfunc] Nlsfunc stays resident in memory once it is initialized. This informational message lets you know that you have already loaded Nlsfunc into memory.

Restore file sequence error

[Restore]

*You have restored files in the wrong order.
You must insert the backup disks in the same order that they were backed up.*

***** Restoring files from drive x: *****

Diskette: n

[Restore]

This message is displayed during the restore process.

Resynch failed. Files are too different

[Fc]

Fc compares what can be loaded into memory. If no lines match in the portion of the files in the buffer space, Fc displays this message.

Same drive specified more than once

[Fastopen]

*You tried to activate Fastopen for the same drive more than once.
There is no need to reactivate it for the same drive.*

Same parameter entered twice

[Format]

This is an informational message indicating that you have entered either /T, /N, /F, or /V twice and need to reenter the correct parameter.

SECOND diskette bad or Incompatible

[Diskcomp]

*The second disk does not contain the same format as the first disk, or Diskcomp does not recognize the format of the second disk.
You should run chkdisk to help you identify the problem.*

Sector not found error reading (or writing) drive x:

[MS-DOS device error]

*This error usually means the disk has a defective spot so that MS-DOS cannot find the requested information on it.
You should copy all files from the disk onto a good disk and then try to reformat the defective disk.*

Sector size too large in file filename

[MS-DOS]

*The specified device driver loaded by config.sys uses a sector size larger than that of any other device driver on the system.
You cannot run this device driver.*

Seek error reading (or writing) drive x:

[MS-DOS device error]

*MS-DOS is unable to locate the information on the disk.
Make sure that the disk is properly inserted in the drive, or try a different drive.*

SHARE already installed

[Share]

Share can only be installed once.

Sharing violation reading drive x:

[MS-DOS device error]

*A program tried to access a file that another program was currently using.
Type A (for Abort), or wait awhile and type R (for Retry).*

SMARTDrive: Computer must be PC-AT or PC-AT compatible

[Smartdrv.sys]

You do not have extended memory because you don't have an IBM PC-AT or PC-AT compatible computer. SMARTDrive program not installed.

If you have expanded memory, use /A on the smartdrv.sys command line. If you have neither extended nor expanded memory, you can't use SMARTDrive.

SMARTDrive: Expanded Memory Manager not present

[Smartdrv.sys]

You included the /A switch in the smartdrv.sys command line but smartdrv.sys could not find the expanded memory manager. Your system boot disk did not install the expanded memory manager. Your config.sys file did not contain the appropriate information.

Consult the documentation for your expanded memory hardware for the correct installation instructions.

SMARTDrive: Expanded Memory Manager Status shows error

[Smartdrv.sys]

While trying to set up SMARTDrive in expanded memory, DOS detected an error. DOS will not install the SMARTDrive program.

Run your expanded memory diagnostics to check your expanded memory. Take the appropriate corrective action as instructed in your expanded memory manual.

SMARTDrive: I/O error accessing drive memory

[Smartdrv.sys]

DOS detected an error while trying to set up SMARTDrive. DOS will not install the SMARTDrive program.

If you want to use the SMARTDrive program, you must add memory to your system.

SMARTDrive: Incorrect DOS version

[Smartdrv.sys]

Smartdrv.sys runs only on 2.x, 3.x, and 4.x versions of DOS. DOS will not install the SMARTDrive program.

SMARTDrive: Insufficient memory

[Smartdrv.sys]

Your system has insufficient memory available for smartdrv.sys. DOS will not install the SMARTDrive program.

If you want to use the SMARTDrive program, you must add memory to your system.

SMARTDrive: Invalid parameter

[Smartdrv.sys]

The command line contains too many parts, such as more than one number or more than one pathname. The size number is out of the range of permitted numbers. For example, you may have the smartdrv.sys size set for 8K, which is too small. DOS will not install the SMARTDrive program.

Edit your config.sys file and change the incorrect smartdrv.sys line.

SMARTDrive: No extended memory available

[Smartdrv.sys]

Your system has no memory available for smartdrv.sys. DOS will not install the SMARTDrive program.

If you want to use the SMARTDrive program, you must add memory to your system.

SMARTDrive: No hard drives in system

[Smartdrv.sys]

Your system has no hard disk. DOS will not install the SMARTDrive program. SMARTDrive only works with hard disks.

If you want to use the SMARTDrive program, you must add a hard disk to your system.

SMARTDrive: Too many bytes per track on hard drive

[Smartdrv.sys]

Your system has a hard disk that smartdrv.sys does not understand. DOS will not install the SMARTDrive program.

SORT: Incorrect DOS version

[Sort]

Sort does not run on MS-DOS versions before 2.0.

SORT: Insufficient disk space

[Sort]

The disk is full.

SORT: Insufficient memory

[Sort]

There is not enough memory to run the Sort program.

Source and target drives are the same

[Backup][Restore]

You specified the same drive for the source and target disks.

Source disk is Non-removable

[Backup]

This is an informational message indicating that the source disk is a hard disk.

SOURCE diskette bad or incompatible

[Diskcopy]

Your disk either has bad sectors or is incompatible with the source drive type (e.g., a high-capacity disk in a 320/360K disk drive).

Source does not contain backup files

[Restore]

You are attempting to restore files from a disk that does not contain backup files.

Source path required

[Replace]

You did not specify a source path for the Replace command.

Specified COMMAND search directory bad

[MS-DOS]

The Shell command in the config.sys file is incorrect.

Make sure that the command.com file exists and that MS-DOS can find it.

Specified drive does not exist, or is non-removable

[Diskcomp][Diskcopy]

You cannot compare or copy hard disks with this command.

You must specify the name of a valid floppy drive.

Strike a key when ready...

[MS-DOS]

This prompt occurs during command processing and is always accompanied by another message. This message is also displayed if you have inserted a Pause command in a batch file. Usually, MS-DOS asks you to insert disks into appropriate drives before this prompt.

To begin command processing, press any character, any number key, the SPACEBAR, or the ENTER key.

Syntax error

[Attrib][Find][MS-DOS]

You have entered a command incorrectly.

Check to make sure you have typed the command correctly and have used appropriate quotation marks.

Syntax errors in GRAPHICS profile

[Graphics]

You have entered a command incorrectly.

Check to make sure you have typed the command correctly.

System transferred

[Format][Sys]

The system files were transferred during Format or Sys command processing.

Target cannot be used for backup

[Backup]

Either the target disk has an unrecognizable format, or it is bad.

You should either try to format the disk with the format command, run chkdsk on it to determine the problem, or not use the disk.

Target diskette bad or Incompatible

[Diskcopy]

Either the target disk has an unrecognizable format, or it is bad.

Try to format the disk with the format command, or run chkdsk on it to determine the problem.

Target diskette may be unusable

[Diskcopy]

Either the target disk has an unrecognizable format, or it is bad.

Try to format the disk with the format command, or run chkdsk on it to determine the problem.

Terminate batch job (Y/N)?

[MS-DOS]

If you press CONTROL+C while in batch mode, MS-DOS asks you whether or not you wish to end batch processing.

Type Y (for Yes) to end processing, or type N (for No) to continue.

The current active keyboard table is xx with code page: yyy

The current active CON code page is zzz

[Keyb]

This is an informational message that shows the current keyboard code, code page for the system, and code page for the console (screen).

The last file was not restored

[Restore]

There was not enough room on the target disk for the file, or the last file was bad.

Use the chkdsk command to determine the problem.

The only bootable partition on drive 1 is already marked active

[Fdisk]

You are trying to change the active partition. The active partition must reside on the first hard disk drive on your system and must be bootable. The only bootable partition on the first hard disk drive is already the active partition.

Too many drive entries

[Fastopen]

You can use Fastopen with up to four hard drives. You have tried to specify a fifth drive.

Too many extent entries

[Fastopen]

The total number of extent entries (m) cannot exceed 999.

Too many file/directory entries

[Fastopen]

The total number of file or directory entries (n) cannot exceed 999.

Too many files open

[Edlin][Label]

MS-DOS could not open the .bak file or write the volume label due to the lack of available system file handles.

Increase the value of the files command in the config.sys file.

Too many name entries

[Fastopen]

The total number of entries specified for a drive exceeded the maximum of 999.

Too many open files

[Backup][Fc][Restore][Xcopy]

MS-DOS could not open the files that you want to compare due to the lack of available system file handles.

Increase the value of the files command in the config.sys file.

Top level process aborted, cannot continue

[Command]

You chose to abort and the error is unrecoverable.

You must use another disk.

Track 0 bad - disk unusable

[Format]

The Format command can accommodate defective sectors on the disk, except for those near the beginning.

You must use another disk.

Tree past this point not processed

[Chkdsk]

This is an informational message indicating that Chkdsk cannot continue processing past this directory path because of the above-mentioned error.

Unable to access Drive x

[Fdisk]

Fdisk cannot access your drive x.

Turn off your system and then restart.

Unable to create directory

[Mkdir][Xcopy]

MS-DOS could not create the directory you specified.

Check to see that there is not a name conflict. You may have a file with the same name, or the disk may be full.

Unable to create KEYB table in resident memory

[Keyb]

MS-DOS tried to create a country-specific table for the keyboard code specified, but failed.

Check the amount of available memory. There may not be enough memory available to create this table.

Unable to perform refresh operation

[Mode]

Printer.sys doesn't have a copy of the code page in its Random Access Memory (RAM) to download to the printer.

You must prepare and select the code page and make sure that the number of buffers (n) in device=printer.sys is equal to or greater than 1.

Unable to reload with profile supplied

[Graphics]

You already loaded Graphics once and didn't allocate enough memory for a second, different profile.

Start your system again.

Unable to shift Screen

[Mode]

Mode is unable to shift the test pattern on the screen any farther.

Unable to write BOOT

[Format]

If the first track of the disk or DOS partition is bad, the Boot record cannot be written on it.

You must use another disk and retry the format command.

Unexpected DOS Error n

[Replace]

An unexpected error n occurred, where n is the MS-DOS error number.

Unrecognized command in CONFIG.SYS

[MS-DOS]

There is an invalid command in your config.sys file.

Refer to Appendix B, How to Configure Your System, for a list of valid statements.

Unrecoverable error in directory

Convert directory to file (Y/N)?

[Chkdsk]

This message is displayed if Chkdsk is unable to correct an error in a directory.

If you respond Y (for Yes) to this prompt, chkdsk converts the bad directory into a file. You can then fix the directory or delete it. If you respond N (for No) to this prompt, you may not be able to write to or read from the bad directory.

Unrecoverable read error on drive x side N, track N

[Diskcomp]

After four attempts, the data still could not be read from the indicated side and track.

Try the command again with a new disk, or copy all files from the damaged disk to a new disk. Either reformat the bad disk or discard it.

Unrecoverable read (or write) error on drive x:

[MS-DOS device error]

MS-DOS is unable to read or write data to the specified device.

Make sure that the disk is properly inserted in the disk drive. Then type R (for Retry). If the error occurs again, type A (for Abort).

usage: fc [/a] [/b] [/c] [/l] [/lb n] [/w]

[t] [/n] [/NNNN] file1 file2

[Fc]

One of the switches that you have specified is invalid.

VERIFY is off (or on)

[MS-DOS]

This message tells you the current setting of the Verify command.

Volume in drive x: has no label

[Dir][Label][Vol]

This is an informational message displayed in response to the Dir, Label, or Vol command.

Volume in drive x: is filename

[Dir][Label][Vol]

This is an informational message displayed in response to the Dir, Label, or Vol command.

Volume label (11 characters, ENTER for none)?

[Format][Label]

This message is displayed when you specify the Label command or the lv switch in the Format command.

Type a volume label, or press the ENTER key to indicate that you do not want a volume label for this disk. It's a good idea, though, to specify a volume label to help you identify your disks.

Volume label does not match

[Fdisk]

You must enter the correct volume label for this drive.

Volume serial number is N-N

[MS-DOS]

This is an informational message.

**WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE x: WILL BE LOST!
Proceed with Format (Y/N)?**

[Format]

This message appears when you try to format a hard disk that already contains data.

If you type Y (for Yes), the data on the disk will be erased. If you do not want the files on your hard disk erased, type N (for No). Copy the files to a floppy disk and repeat the format command.

Warning - directory full

[Recover]

The root directory is too full for Recover processing.

Delete some files in the root directory to free space for the recovered files, and try the command again.

**Warning! Diskette is out of sequence
Replace diskette or continue if okay
Strike any key when ready**

[Restore]

You should restore the diskettes in the order that you backed them up.

**Warning! File filename is a hidden (or read-only) file
Replace the file (Y/N)?**

[Restore]

This message prompts you as to whether you want to replace a hidden or read-only file.

Type Y (for Yes) if you want to restore the hidden or read-only file from the backup disk. Type N (for No) if you do not want to restore this file.

**Warning! File filename was changed after it was
backed up
Replace the file (Y/N)?**

[Restore]

This message prompts you as to whether you want to replace a backup file that has been changed.

Type Y (for Yes) if you want to restore this file, or type N (for No) if you do not.

**WARNING! File X is a read-only file
Replace the file (Y/N)?**

[Restore]

This message prompts you as to whether you want to replace a read-only file.

Type Y (for Yes) if you want to restore the read-only file from the backup disk. Type N (for No) if you want to restore this file.

**Warning! Files in the target drive
\\Backup (or root) directory will be erased**

[Backup]

Backup found files in the target drive, and you did not specify the /a switch to append files.

Warning! No files were found to back up

[Backup]

Backup did not find any files to back up on the disk you specified.

Warning! No files were found to restore

[Restore]

Restore did not find the file that you wanted to restore from the backup disk.

Warning: Read error in EXE file

[Exe2bin]

The amount read was less than the size of the header. This is a warning message only.

Warning! The partition marked active is not bootable

[Fdisk]

The active DOS partition must be bootable.

Write failure, diskette unusable

[Sys]

This message indicates that a DOS critical error occurred when Sys was writing data to the disk.

You must rerun the sys command with a different disk.

Write fault error writing drive x:

[MS-DOS device error]

MS-DOS is unable to write data to the specified device.

Make sure that the disk is properly inserted in the disk drive. Then type R (for Retry). If the error occurs again, type A (for Abort).

**Write protect error
Format terminated**

The disk you are trying to format is write protected.

Write protect error writing drive x:

[MS-DOS device error]

You tried to write data on a write-protected disk.

If the disk has a write-protect tab on it, you must remove the tab before you can write on the disk. (You should consider first why the disk was write-

Index

- > prompt 111
- \$ prompt 111
- .bak extension 175-176
- .BAK extension 198
- .bat extension 16-17, 143, 149
- .com extension 16, 236
- .doc extension 53, 115
- .exe extension 16, 208-209, 236
- .lib extension 209
- .map extension 209
- .obj extension 208
- .ref extension 53
- .txt extension 9, 10, 53, 115
- /cparmaxalloc option, abbreviation 225
- /cparmaxalloc option, link 224-225
- /dosseg option, abbreviation 229
- /dosseg option, link 228-229
- /dsallocate option, abbreviation 226
- /dsallocate option, link 217, 226
- /exepack option, abbreviation 220
- /exepack option, link 220
- /help option, abbreviation 219
- /help option, link 219
- /high option, abbreviation 225
- /high option, link 217, 225
- /linenumbers option, abbreviation 222
- /linenumbers option, link 211, 221-222
- /map option, abbreviation 221
- /map option, link 211, 216, 221
- /nodefaultlibrarysearch option, abbreviation 223
- /nodefaultlibrarysearch option, link 223
- /nogrouppassociation option, abbreviation 226
- /nogrouppassociation option, link 226
- /noignorecase option, abbreviation 223
- /noignorecase option, link 222
- /overlayinterrupt option, abbreviation 227
- /overlayinterrupt option, link 227
- /p switch 109, 279
- /pause option, abbreviation 219
- /pause option, link 217, 219-220
- /segments option, abbreviation 228
- /segments option, link 228
- /stack option, abbreviation 224
- /stack option, link 223-224
- < prompt 111
- Abbreviation
 - /cparmaxalloc option, link 225
 - /dosseg option, link 229
 - /dsallocate option, link 226
 - /exepack option, link 220
 - /help option, link 219
 - /high option, link 225
 - /linenumbers option, link 222
 - /map option, link 221
 - /nodefaultlibrarysearch option, link 223
 - no grouppassociation option, link 227
 - /noignorecase option, link 222
 - /overlayinterrupt option, link 227
 - /pause option, link 219
 - /segments option, link 228
 - /stack option, link 224
 - debug memory locations
 - BYTE 242
 - WORD 242
- Aborting a command 171
- Accounts directory 6
- Adding a file 116-118
- Adding lines, Edlin 186
- Adding spacing to batch files 162
- Align type, link 230
- Allocate expanded memory command, debug
 - abbreviation 270
 - comments 270
 - purpose 270
 - syntax 270
- Allocating a data group, link 226
- Allocating disk space 2, 63-64
- Alphabetical sorting 19, 151-152
- Alphabetizing a file 128-129, 151-152
- American National Standards Institute (ANSI) 298
- ANSI escape sequence
 - CUB 299
 - CUD 299
 - CUF 299
 - CUP 299
 - CUU 299
 - definition 298
 - driver 112
 - DSR 299
 - ED 300

- EL 300
- HVP 299
- modes of operation 301
- RCP 300
- RM 301
- SCP 299
- SGR 300
- SM 301
- using in config.sys file 286
- variables 298
- Ansi.sys
 - device driver 286, 297-304
 - installing 298
- Append command
 - notes 29-31
 - purpose 29
 - switches 30
 - syntax 29
- Append command, Edlin
 - purpose 186
 - syntax 186
- Appending files
 - Copy command 51, 53
- Appending output 18
- Archive attribute
 - Xcopy
 - switches 139
- Arglist
 - definition 237
- Argument 25
- ASCII characters 245
- ASCII code prompt (escape) 111, 112
- ASCII file
 - Copying 51
- Assemble command, debug
 - abbreviation 242
 - comments 242
 - notes 32-33
 - purpose 242
 - syntax 242
- Assign command
 - purpose 32
 - syntax 32
- Asterisk (*)
 - as system prompt in Edlin 174-175, 183
 - as wildcard character 9-10
- Asynchronous communications (COM)port 99
- Attrib command
 - notes 34-35
 - purpose 34
 - syntax 34
- Attribute
 - setting 139
- Attribute, setting 34-35
- Autoexec.bat file
 - creating 122-133, 145-148
 - and Date command 55, 146
 - defined 145-146
 - options
 - and Path command 18
 - and Set command 124
 - setting system code pages in 330
 - updating 122
- Background color functions, ANSI 301
- Backslash (\) 7-8
- Backup
 - and reconfiguring your disk 315
 - read-only 176, 191
- Backup command
 - backup log file 36
 - and country code 284
 - disk sizes 37
 - exit codes 38
 - notes 37-38
 - purpose 36
 - and Restore command 37, 119-120
 - switches
 - /a: 36
 - /d: 36
 - /f: 36
 - /L: 36
 - /m 36
 - /s 36
 - /t: 36
 - syntax 36
- Backup disk 2
 - disk sizes 37
 - labeling and numbering 37
- Backup log file 37
- Bad disk sectors 113-114
- BASIC See GW-BASIC
- Batch commands. See Batch processing commands
- Batch file
 - checknew.bat 144, 154
 - comment 162
 - creating 143-144
 - definition 143

- dividing into pieces 161
- label 158
- readability and spacing 162
- remark 162
- running 144-145, 151
- sorter.bat 149, 151-152
- stopping 161
- suspending execution of 161
- using a colon in 158
- using named parameters in 150-151
- using a percent sign in 149, 156-157
- using redirection symbols in 145
- using replaceable parameters with 124, 149
- using temporary files with 152
- using the shift command with 163
- Batch processing See Batch file; Batch processing commands.
- Batch processing commands 153-163
 - Call 153-154
 - Echo 153, 155
 - For 153, 156-157
 - Goto 153, 158
 - If 153, 159-160
 - Pause 153, 161
 - Rem 153, 162
 - Shift 153, 163
- Batch program, halting 161
- Baud rate 95, 99
- Bin directory 6, 17
- Binary file
 - copying 51
 - displaying 135
- Binary files
 - comparing binary files 73
- Binary format 66-67
- Blink graphics function, ANSI 300
- Block devices
 - Drivparm command 278, 287-288
- Block of text, moving, Edlin 195-196
- Bold graphics function, ANSI 300
- Brackets, notation used for command options 25
- Break command, MS-DOS 39
- Break command, config.sys
 - definition 278
 - purpose 280
 - syntax 280
- Buffer space (Fc) 72
- Buffer, internal 108
- Buffers command, config.sys file
 - comments 281-282
 - definition 278, 279
 - purpose 281
 - syntax 281
- Byte 239
- Bytes, available 44
- Call command, batch 153, 154
- Capital letters, use of 222-223, 237
- Capitalization
 - Country command 278, 285
- Caret (^) 237
- Case conversion
 - Country command 283
- Case-sensitivity, preserving 222
- Changing directories 11-12, 42-43
- Changing the name of a file 115
- Changing the time format 132
- Chcp command 330
 - purpose 40
 - syntax 40
- Chdir command
 - changing the working directory 11-12
 - displaying the working directory 11-12, 42-43, 58-59
 - purpose 42
 - shorthand notation 12, 42-43
 - syntax 42
- Checking a disk 3, 44-45
- Checknew.bat file 144, 154
- Chkdsk command
 - checking files for errors 3, 44
 - purpose 3, 44
 - recovering files 113-114
 - status report 3, 44-45
 - switches
 - /f 44-45
 - /v 45
 - syntax 44
- Class type, link 230, 232
- Clearing the screen 46
- Cls command
 - purpose 46
 - syntax 46
- Code-page information (.cpi) file 102, 327
- Code-page switching 102, 123
- Code pages
 - Chcp command 40-41
 - Chcp messages 40-41"
 - definition 327

- displaying current code pages 102-103, 333
- for a specific country 283-285
- how to use 327-334
- refreshing lost code pages 103, 334
- setting country and keyboard codes 330-331
- setting device code pages 102-103
- setting parallel printer code pages 102-103, 332
- setting screen code pages 102, 331
- setting the system code page 329-330
- switching between code pages 333
- using hardware and prepared code pages 327, 331
- See also Graftabl command.
- Code Segment: Instruction Pointer (CS:IP) 66
- Collating sequence
 - Country command 278
- Colon, using in a batch file 158
- Color graphics adapter (CGA) 81
- Column parameter 298
- COM. See Asynchronous communications port.
- Combine types, link
 - at 231-232
 - common 231
 - memory 231
 - private 232
 - public 231
 - stack 231
- Combining files 53
- Combining segments, link 231-233
- Command
 - Append 29
 - Assign 32
 - Attrib 34
 - Backup 36, 120
 - batch processing 153
 - Break 39, 278, 280
 - Buffers 278, 281
 - Call 154
 - Chcp 40
 - Chdir 11, 42
 - Chkdsk 3, 44-45, 113-114
 - Cls 46
 - Command 47
 - commands not usable over network
 - Chkdsk 22
 - Diskcomp 22
 - Diskcopy 22
 - Fastopen 22
 - Fdisk 22
 - Format 22
 - Join 22
 - Label 22
 - Recover 22
 - Subst 22
 - Sys 22
- Comp 49
- conditional execution 159
- Copy 16, 51
- Country 278, 283
- Ctty 54
- Date 55
- debug commands 238
- default value 24
- Del 10, 16, 57
- Device 278, 286
- Dir 7, 10, 12, 16, 58
- Diskcomp 60
- Diskcopy 16, 63
- diskcopy.exe 16
- Drivparm 278, 287
- Echo 155
- Exe2bin 66
- Exit 68
- external 15, 16, 17
- Fastopen 69
- Fc 71
- FCBS 278, 290
- Fdisk 74
- Files 278, 291
- Find 19, 75
- For 156
- Format 3, 16, 77
- format.bat 16
- Goto 158
- Graftabl 81
- Graphics 83
- If 159
- Install 278, 292
- internal 15
- Join 85
- Keyb 87-89
- Label 90
- Lastdrive 278, 293
- list of commands 26-28
- Mem 92
- Mkdir 11, 94
- Mode 95
- More 19, 105
- Nlsfunc 106

- Path 17, 107
- Pause 161
- pipeline 20
- pipng 19
- Print 108
- Prompt 111
- Recover 113
- Rem 162, 278, 294
- Ren 115
- repeating 165
- Replace 116
- Restore 119
- Rmdir 12, 121
- rules for using options
 - brackets 25
 - ellipsis 25
 - equal sign 25
 - italic 25
 - semicolon 25
 - spaces and tabs 25
- sample command page 22
- Select 122
- Set 124
- Share 126
- Shell 278, 294
- Shift 163
- Sort 19, 128
- Stacks, config.sys file 278, 296
- Subst 130
- symbol
 - non-network 22
- symbol of external 22
- symbol of internal 22
- syntax 24
- Sys 131
- Time 132
- Tree 134
- Type 16, 135
- Ver 136
- Verify 137
- Vol 138
- Xcopy 139
- Command command
 - purpose 47
 - switch
 - /c 47
 - /e 47
 - /p 47
 - syntax 47
- Command information, argument 25
- Command line
 - parameters, changing position of 163
 - relation to template 166
- Command lines
 - with Link 210
- Command options, Edlin 184
- Command parameters, debug
 - address 239
 - byte 239
 - count 241
 - description 239
 - drive: 239
 - handle 241
 - list 240
 - lpage 241
 - ppage 241
 - range 240
 - record 239
 - string 240
 - value 239
- Command processor, MS-DOS 47, 68
- COMMAND.COM 15, 159
- command.com 278
- Command.com 279
- COMMAND.COM
 - exiting 68
- Comment parameter
 - Pause command 161
 - Rem command 162
- Comment, batch file 162
- Communications port
 - COM1 99
 - COM2 99
- Comp command
 - purpose 49
 - syntax 49
- Compare command, debug
 - abbreviation 244
 - comments 244
 - purpose 244
 - syntax 244
- Comparing files 49, 71
- Compilers 221
 - overlays 211, 227
- Computer memory 15
- Computer network 22
- COMSPEC
 - definition 295

- Concatenation 53
- Concealed function, ANSI 300
- Condition parameter 159
- Conditional execution of commands 159
- config.sys command
 - Break 278
- Config.sys command
 - Break 280
 - Buffers 278, 281
 - Country 278, 283
 - Device 278, 286
 - Driveparm 287
 - Drivparm 278
 - FCBS 278, 290
 - Files 278
- config.sys command
 - Files 291
- Config.sys command
 - Install 278, 292
 - Lastdrive 278, 293
 - Rem 278, 294
 - Shell 278, 294
 - Stacks 278, 296
- config.sys file 277
- Config.sys file
 - creating 122
 - setting system code pages 329
 - updating 122
- Config.sys. example file 278
- Configuration commands
 - listing of 278
- Configuration file
 - definition 277
- Configuring a hard disk 74, 316
- Configuring your system 277
- Console 144
- Console screen
 - code page switching 100, 102
- Console screen device (CON) 88
- Control character functions 170
- Control key sequence
 - CONTROL+C 235
 - CONTROL+S 235
- CONTROL key sequences
 - stop a command, CONTROL+C 170
 - stop screen scrolling, CONTROL+S 170
- CONTROL+ALT+F1 88
- CONTROL+ALT+F2 88
- CONTROL+C 144, 161, 170, 171, 183, 192, 235, 279, 280
 - extending its functions 280
- CONTROL+C check 39, 280
- CONTROL+H 171
- CONTROL+J 171
- CONTROL+N 171
- CONTROL+P 171
- CONTROL+S 170, 171, 235
- CONTROL+V 183
- CONTROL+X 171
- CONTROL+Z 144, 148, 149, 199
- Copy command 143
 - purpose 51
 - switch
 - /a 51
 - /b 51
 - /v 51
 - syntax 51
 - using wildcard with 53
 - versus Diskcopy command 64
 - versus Xcopy command 64
- Xcopy command 52
- Copy command, Edlin
 - purpose 187
 - syntax 187
- Copying
 - directories 140
 - disks 63
 - File Allocation Table 3
 - files 51, 140
 - lines, Edlin 187
 - subdirectories 139
 - template characters 170
- Count option
 - Edlin Copy command 187
- Country (or language)
 - supported by MS-DOS 4.0 283
- Country
 - loading country-specific information 106
 - specifying with Select command 122
- Country command
 - changing date format 55
- Country command, CONFIG.SYS file
 - changing time format 133
- Country command, config.sys file
 - changing time format 133
 - definition 278
 - purpose 283
 - syntax 283

- Country configuration command 329
- Country support
 - MS-DOS 4.0 enhancements to viii
- Country.sys
 - device driver 286
- Create Primary DOS partition menu 318
- Creating a directory 11
- CTRL+Z 52
- CTRL-BREAK 97
- City command
 - purpose 54
 - syntax 54
- CUB-Cursor Backward, ANSI escape sequence 299
- CUD-Cursor Down, ANSI escape sequence 299
- CUF-Cursor Forward, ANSI escape sequence 299
- CUP-Cursor Position, ANSI escape sequence 299
- Currency
 - Country command 278, 283
- Currency symbols 122
- Cursor
 - in the template 168
 - location, Edlin 177, 178, 179, 180
 - movement 298
- CUU-Cursor Up, ANSI escape sequence 299
- Data group, allocating in link 226
- Data path, NUL 30
- Data, losing 335
- Databit 99
- Date
 - Country command 278, 283
- Date command
 - country code 284
 - internal clock 55
 - purpose 55
 - syntax 55
- Date format 122
 - changing 55
 - table, relation to different countries 284
- Xcopy 139
- Date message 56
- Date prompt 111
- Dead keys
 - definition 88
- Deallocate expanded memory command, debug
 - abbreviation 271
 - comments 271
 - purpose 271
 - syntax 271
- Debug
 - arglist 237
 - command parameters
 - address 239
 - byte 239
 - count 241
 - description 239
 - drive: 239
 - handle 241
 - list 240
 - lpage 241
 - ppage 241
 - range 240
 - record 239
 - string 240
 - value 239
 - commands, listing of 238
 - definition 235
 - error indication 237
 - error messages 274
 - hyphen prompt 236
 - illegal ranges 240
 - illegal strings 241
 - memory locations, abbreviating 242
 - mnemonics 242
 - opcodes 243
 - quitting 261
 - starting 236
 - versus Edlin 235
- Debug command
 - Allocate expanded memory 270
 - assemble 242
 - compare 244
 - Deallocate expanded memory 271
 - dump 245
 - enter 247
 - fill 249
 - Get expanded memory status 273
 - go 250
 - hex 252
 - input 253
 - load 254
 - Map expanded memory pages 272
 - move 256
 - name 257
 - output 259
 - proceed 260
 - quit 261
 - register 262

- search 265
- trace 266
- unassemble 267
- write 269
- Debug errors
 - BF - Bad flag 274
 - BP - Too many breakpoints 274
 - BR - Bad register 274
 - DF - Double flag 274
- Debugging programs 235
- Decimal separator
 - Country command 278, 283
- Default code page
 - United States 327
- Default configuration settings
 - changing 277
- Default drive 24, 275
- Default drive prompt 111
- Default filename extension, .lib 209
- Default filename extension, .map 209
- Default filename extension, link 208, 209
- Default libraries, ignoring 223
- Default printer, PRN 108
- Default value, commands 24
- Del command 10
 - purpose 57
 - switch
 - /p 57
 - syntax 57
- DEL command, Edlin
 - purpose 178
 - similarity to F1 178
 - syntax 178
- Delete command, Edlin
 - purpose 188
 - syntax 188
- Deleting
 - a directory 12, 121
 - files 13, 57
 - lines, Edlin 188
 - template characters 170
 - text, Edlin 173
- Device
 - changing the command input and output device 54
 - setting device code pages 102
- Device command, config.sys file
 - definition 278
 - purpose 286
 - syntax 286
- Device configuration command 329
- Device driver 279
 - definition 297
 - Device command 278
 - installable 297
- Device drivers
 - installable 286
- Device error
 - Bad call format error 339
 - Bad unit error 340
 - Not ready error 384
 - Read fault error 390
 - Sector not found error 393
 - Seek error 393
 - Sharing violation 393
- Device, operation mode 95
- Devices
 - using prepared code pages to switch from one language 328
- Difference reporting, (Fc) 72
- Dir command 10
 - checking for config.sys file 277
 - purpose 58
 - switch
 - /p 58
 - /w 58
 - syntax 58
 - using wildcards with 58
- Directives, link 229
- Directories
 - finding directory names
 - Tree 134
- Directory
 - accounts 6
 - bin 17
 - changing 11, 42
 - creating 11
 - deleting 12
 - displaying 12, 58
 - erasing 12
 - external commands, bin 6
 - hierarchical structure 4, 5
 - listing of 7
 - making 94
 - maximum capacity 4
 - memos 6
 - multilevel 3, 4, 5, 7
 - programs 6
 - removing 12, 121

- renaming 13
- root 3, 5, 86
- shorthand notation 7
- sorted listing 129
- sorting 20
- subdirectories 4, 5, 6, 12
 - naming 12
- system 5
- user 6
- working 5, 11, 17, 30
- Disk
 - backing up 36
 - checking 44
 - copying 63
 - disk sizes 37
 - formatting 77
 - fragmented 63
 - media 36, 119
 - sector 113
 - type and size specifiers 78
 - types 36, 119
 - volume label 90, 138
 - write-protected 105
 - writing to 137
- Disk buffer
 - default number of 281
 - definition 281
- Disk buffers 281
- Disk drive 24
 - electrically compatible 288
 - if you have only one 26, 275
 - source 24
 - target 24
 - virtual 130
- Disk error
 - Abort 335
 - Fail 335
 - FCB unavailable 361
 - General failure error 365
 - Ignore 335
 - Invalid disk change error 372
 - Lock violation error 377
 - No paper error error 380
 - Non-DOS disk error 383
 - Retry 335
 - Write fault error 403
 - Write protect error 404
- Disk space, allocation of 63
- Disk swapping 219
- Disk volume label 78
- Disk, hard
 - MS-DOS 4.0 support of viii
- Disk-caching programs 312
- Diskcomp command
 - exit codes 61
 - purpose 60
 - syntax 60
- Diskcopy command
 - option
 - source drive 63
 - target drive 63
 - purpose 63
 - syntax 63
- Diskcopy commandd
 - exit codes 64
- Disks
 - comparing files on different disks 49
- Display
 - setting display mode 100
- Display modes viii
- DISPLAY.SYS 297
- Display.sys
 - definition 305
 - device driver 286
 - installing 305
 - options 305
- Displaying a command, MS-DOS 155
- Displaying a directory 58
- Displaying a file 19, 135
- Displaying lines, Edlin 194
- Displaying output 100
- Displaying the current time 132
- Dividing a batch file into pieces 161
- DOS partition 315
 - active partition
 - definition 321
 - creating 317
 - extended DOS partition 319
 - more than one partition 319
 - primary partition 318
 - single partition 318
- DOS Partition
 - deleting 322
- DOS partition
 - displaying data 324
 - setting the active partition 321
- Drive
 - source 24

- target 24
- Drive letter 130, 138
- Drive name 24
- DRIVER.SYS 297
- Driver.sys
 - definition 306
 - device driver 286
 - installing 306
 - switches 306
- Drivparm command (Config.sys file) 287
- Drivparm command, config.sys file
 - comments 287
 - definition 278
 - purpose 287
 - syntax 287
- DSR-Device Status Report, ANSI escape sequence 299
- Dummy parameters 149
- Dump command, debug
 - abbreviation 245
 - comments 245
 - purpose 245
 - syntax 245
- Echo command, batch
 - purpose 155
 - syntax 155
- Echoing
 - definition 155
 - preventing 155
- ED-Erase Display, ANSI escape sequence 300
- Edit command, Edlin
 - purpose 189
 - syntax 189
- Editing a file 173
- Editing commands
 - Copy multiple characters 177
 - Copy one character 177
 - Copy template 178
 - Create new template 180
 - Enter insert mode 180
 - Quit 179
 - Skip multiple characters 178
 - Skip one character 178
- Editing key
 - description 176
 - DEL 167
 - ESC 167
 - F1 166
 - F2 166
 - F3 166
 - F4 167
 - F5 167
 - F6 167
 - INS 167
- Editing keys 165
- Editing large files in stages 186, 205
- Editing text, Edlin 189
- Edlin
 - .bak extension 175
 - adding lines 186
 - adjusting lines 174
 - asterisk (*) prompt 174
- EDLIN
 - changing your config.sys file with 277
- Edlin
 - command option
 - line 184
 - question mark 185
 - text 185
 - consecutive line numbering 174
 - copying lines 187
 - creating a new file 174
- EDLIN
 - current line, asterisk (*) 183
- Edlin
 - cursor location 177, 178, 179, 180
 - default values 194
 - deleting lines 173, 188
 - displaying lines 173
 - Edit command 183
 - editing an existing file 175
 - editing text 173, 189
 - freeing memory 175
 - insert mode 177, 178, 179, 180
 - inserting text 173, 192
 - line numbers 174
 - listing text 194
 - loading files into memory 175, 186, 205
 - memory capacity 186, 205
 - merging files 204
 - moving lines 195
 - omitting options 194
 - paging through files 197
 - question mark option 202
 - quitting 175, 198
 - renumbering lines 174
 - replace mode 180

- replacing text 199
- revising 173
- saving a file 175, 191
- searching for text 202
- starting 174
- using commands with 182
- writing to a disk 205
- Edlin command
 - Append lines 186
 - Copy lines 187
 - Delete lines 188
 - Edit line 189
 - End editing 191
 - Insert text 192
 - List text 194
 - Move text 195
 - Page 197
 - Quit 198
 - Replace text 199
 - Search text 202
 - Transfer text 204
 - Write lines 205
- Edlin commands
 - listing of 182
- Edlin, line editor 11
- EDLIN, line editor 143
- Edlin, line editor 173
- EL-Erase Line, ANSI escape sequence 300
- Ellipsis, used for options 25
- End-of-file mark 50, 52
- End/Save command, Edlin
 - purpose 191
 - syntax 191
- Ending an edit session, Edlin 191
- Enter command, debug
 - abbreviation 247
 - comments 247
 - purpose 247
 - syntax 247
- ENTER-LINEFEED prompt 111
- Environment 150
- Environment strings 145
- Environment variable
 - LIB 215
- Environment, definition 124
- Equal sign, used for options 25
- Erase 57
- Erasing
 - files 13
 - the screen 300
- Error checking
 - directories 3
 - disks 3
 - File Allocation Table 3
 - files 3
- Error indication, debug 237
- Error message
 - action 335
 - device 335
 - disk 335
 - type 335
- Error message, network 22
- Error messages
 - debug 274
- Error messages, 'MS-DOS 335
- Error, recovering from 335
- Error, responding to
 - Ignore 335
- ESC command, Edlin
 - purpose 179
 - similarity to F5 179
 - syntax 179
- Escape sequence, ANSI 112, 286, 298, 300
- Example config.sys file 278
- Exe2bin command
 - purpose 66
 - syntax 66
- Executable commands
 - Path command 107
- Executable file
 - creating 207
- Executable file packing, link 220
- Executable file, .exe 16
- Executable image 229
- Executable program, link 210
- Executing a .bat File 151
- Exit code 159
- Exit command
 - purpose 68
 - syntax 68
- Expanded memory
 - displaying 92
- Expanded Memory Specification (EMS) memory 92
- Expanded memory support 123
- Extended DOS partition 319
- Extended memory
 - displaying 92
- Extended memory support 123

Extension

- .bak 175
- .BAK 198
- .bat 16
- .BAT 143
- .bat 149
- .com 16, 236
- .doc 53
- .DOC 115
- .exe 16, 209, 236, 255
- .lib 209
- .map 209
- .obj 208
- .ref 53
- .TXT 9
- .txt 53
- .TXT 115
- substitution 53

External command

- bin directory 6
- definition 16
- list of commands 17
- searching for 107
- symbol 22

F1 command, Edlin

- purpose 177
- similarity to DEL 177
- syntax 177

F2 command, Edlin

- purpose 177
- similarity to F4 177
- syntax 177

F3 command, Edlin

- purpose 178
- syntax 178

F4 command, Edlin

- purpose 178
- similarity to F2 179
- syntax 178

F5 command

- similarity to ESC 180

F5 command, Edlin

- purpose 180
- syntax 180

Fastopen command

- config.sys processing 292
- purpose 69
- syntax 69

Fc command

- purpose 71
- syntax 71

Fc switches 71**FCB unavailable 361****FCBS command, config.sys file**

- comments 290
- definition 278
- purpose 290
- syntax 290
- when to use 290

Fdisk 315**Fdisk command**

- menus 74
- purpose 74
- starting 316
- syntax 74

Fdisk menu**option**

- creating primary DOS partition 318
- deleting a DOS partition 322
- selecting the next fixed disk drive 326
- setting the active partition 321

Fdisk utility

- checking for previously configured and formatted disk 316

File**alphabetizing 128****ASCII 51****attribute 34****autoexec.bat 18, 55, 145, 148****backup 198****backup log files 37****BACKUP.LOG 36****batch 124, 149****binary 51****COMMAND.COM 15****concatenating 53****CONFIG.SYS 133****config.sys 133, 277****conversion to binary format 66****copying 51****creating a new file, Edlin 174****deleting 57****deleting a temporary file 152****displaying 19, 135****editing 173****editing an existing file, Edlin 175****grouping 4**

- hidden 44, 119
- location of 2
- locking 126
- organization 3, 7
- printing 108
- protection 2
- read-only 116, 119
- recovery 113
- renaming 115
- searching for 5
- separation by category 3
- sharing 126
- sorter.bat 149
- sorting 19, 128
- system 4
- temporary 105, 149, 152
- text 173
- updating 173
- File Allocation Table
 - allocating disk space 2
 - copying of 2
 - definition 2
 - locating files 2
- File command, config.sys file
 - comments 291
- File handles
 - versus file control blocks in accessing files 290
- File header, MS-DOS 218
- File restoring 119
- Filename
 - as a command option 24
 - autoexec.bat 18
 - COMMAND.COM 15
 - syntax 24
- Filename extension
 - .bat 16
 - .com 16
 - .exe 16, 209
 - .lib 209
 - .map 209
 - .obj 208
 - .TXT 9
 - when to add 17
- Filenames
 - Comp command 49
- Files
 - comparing 49, 71
 - comparing files of different sizes 50
 - comparing files with mismatching information 49
 - viewing long files 105
- Files command, config.sys file
 - definition 278
 - purpose 291
 - syntax 291
- Fill command, debug
 - abbreviation 249
 - comments 249
 - purpose 249
 - syntax 249
- Filter
 - definition 19
- Find 19
- More 19
- More command 105
- Sort 19, 129
- Sort command 128
- Find command
 - purpose 75
 - switch
 - /c 75
 - /n 75
 - /v 75
 - syntax 75
- Finding a file 5
- Finding directory names 134
- Fixups, link 232
- Flags 263
- Floppy disk
 - installing 'MS-DOS 122
- Floppy disk drive, if you have only one 275
- Folding format
 - Country command 278, 283
- For command, batch
 - purpose 156
 - syntax 156
- Foreground color functions, ANSI 300
- Format command
 - /s switch 318
 - purpose 77
 - switches 77
 - syntax 77
 - use after configuring a hard disk 316
- Formatting a disk 77
- Formatting a hard disk 79
- Fragmented disk 63
- Frame number, canonical 230
- Freeing memory, Edlin 175
- Function key

- description 176
- DEL 167
- ESC 167
- F1 166
- F2 166
- F3 166
- F4 167
- F5 167
- F6 167
- INS 167
- Functions, special editing 166
- General failure error 365
- Get expanded memory pages command, debug
 - abbreviation 273
 - comments 273
 - purpose 273
 - syntax 273
- Go command, debug
 - abbreviation 250
 - comments 250
 - purpose 250
 - syntax 250
- Goto command, batch
 - purpose 158
 - syntax 158
- Graftabl command
 - purpose 81
 - switches 81
 - syntax 81
- Graphic functions, changing 298
- Graphics command
 - printer options 83
 - purpose 83
 - switches 84
 - syntax 83
- Graphics command viii
- Graphics monitor
 - color disabled (BW) 100
 - color enabled (CO) 100
- Graphics, ANSI escape sequences
 - function
 - Background colors 300
 - Blink 300
 - Bold 300
 - Concealed 300
 - Foreground colors 300
 - Reverse Video 300
 - functions 300
 - parameters 300
 - Set Graphics Rendition-SGR 300
- Grouping files 4
- Groups, linking 232
- Halting a command 171
- Hard disk
 - configuration 74
 - configuring 315
 - formatting 316, 318
 - installing 'MS-DOS 122
- Hardware code page 327
- Hex command, debug
 - abbreviation 252
 - comments 252
 - purpose 252
 - syntax 252
- Hidden file 44, 131
- Hidden file, restoring 119
- Hidden files
 - Rmdir command 121
- Hierarchical directory, definition 4, 6
- High start address, setting 225
- HVP-Horizontal and Vertical Position, ANSI escape sequence 299
- Hyphen prompt, debug 236
- Icon
 - internal command icon 153
- If command, batch
 - purpose 159
 - syntax 159
- Illegal ranges, debug 240
- Illegal strings, debug 241
- Initializing
 - Select command 122
- Input 18
- Input command, debug
 - abbreviation 253
 - comments 253
 - purpose 253
 - syntax 253
- Input, redirecting 19
- INS command, Edlin
 - purpose 180
 - syntax 180
- Insert command, Edlin
 - purpose 192
 - syntax 192

- Insert mode, Edlin 177, 178, 179, 180
- Inserting template characters 170
- Inserting text, Edlin 173, 192
- Install command, config.sys file
 - definition 278
 - purpose 292
 - syntax 292
- Installation options
 - specifying with Select command 123
- Installing MS-DOS
 - Select command 122
- Interactive processing 156
- Internal buffer 72, 108
- Internal clock 133
 - Date command 55
- Internal command 15, 153
 - definition 15
 - list of commands 15
 - symbol 22
- International
 - case conversion 283
 - currency 283
 - date 283
 - time 283
- Interrupt numbers 227
- Invalid disk change error 372
- Inverse video mode 112
- Italics, used for options 25
- Join command
 - purpose 85
 - reversing the command 86
 - syntax 85
- Joining files 53
- Key
 - editing 165
 - LINEFEED 171
 - s-2INS 167
 - template 165
 - DEL 167
 - ESC 167
 - F1 166
 - F2 166
 - F3 166
 - F4 167
 - F5 167
 - F6 167
- Key sequence
 - CONTROL+C 171, 183, 192, 280
 - CONTROL+H 171
 - CONTROL+J 171
 - CONTROL+N 171
 - CONTROL+P 171
 - CONTROL+S 171
 - CONTROL+V 183
 - CONTROL+X 171
 - CONTROL+Z 199
- Keyb command 87, 330
 - purpose 87
 - syntax 87
- Keyboard
 - changing keyboard programs 89
 - copying information from 144
 - MS-DOS 4.0 - supported keyboards 88
 - QWERTY format 88
 - setting typematic rate 101
 - specifying with Select command 122
 - type 87
- Keyboard command
 - config.sys processing 292
- Keyboard.sys
 - device driver 286
- Keys, reassigning 298
- Label
 - for logical or physical disk drives 279
- Label command 90
 - purpose 90
 - syntax 90
- Label variable, batch 158
- Language (or country)
 - supported by MS-DOS 4.0 283
- Lastdrive command, config.sys file
 - definition 278
- Lastdrive, config.sys file 293
 - comments 293
 - network environment 293
 - purpose 293
 - syntax 293
- Letters, lowercase 237
- Letters, uppercase 237
- LIB
 - environment variable 215
- Library file 211
- Library files 209, 215
- Line editor, Edlin 11
- Line editor, EDLIN 143

- Line editor, Edlin 173
- Line mode 301
- Line option, Edlin
 - description 184
 - number 184
 - number sign 184
 - period 184
- Line parameter 298
- LINEFEED key 171
- Lineprinter, output to 171
- Link
 - /linenumbers 211
 - align type 230
 - canonical frame number 230
 - class type 230
 - command line 210
 - creating an executable file 210
 - default extensions 208
 - description 207
 - directives 229
 - frame number, canonical 230
 - groups 232
- link
 - library files 209
- Link
 - library files 211, 215
 - library search 223
 - map file 209, 211, 216, 221
 - operation 229
 - option
 - /dsallocate 217
 - /high 217
 - /map 211, 216
 - overlays 211, 227
 - placeholder 210
 - preserving case-sensitivity 222
 - prompts 208
 - references
 - long 232
 - near segment-relative 232
 - near self-relative 232
 - short 232
 - response file 213
 - search paths 215
 - starting 208
 - temporary file 217
 - using the comma with 209
 - using the semicolon with 209
 - variable
 - executable file 210
 - libraryfiles 211
 - mapfile 211
 - objectfiles 210
- Link option /ndefaultlibrarysearch 223
- Link option /noignorecase 222
- Link option
 - /cparmaxalloc 224
 - /dosseg 228
 - /dsallocate 226
 - /exepack 220
 - /help 219
 - /high 225
 - /linenumbers 221
 - /map 221
 - /nogroupassociation 226
 - /overlayinterrupt 227
 - /pause 219
 - /segments 228
 - /stack 223
 - abbreviations for 218
- Link options
 - listing of 218
- Linking
 - described 207, 229
- List command, Edlin
 - purpose 194
 - syntax 194
- Listing a directory 7
- Listing lines, Edlin 194
- Load command, debug
 - abbreviation 254
 - comments 254
 - purpose 254
 - syntax 254
- Locating a file 5
- Locating files 2
- Lock violation error 377
- Locking a file 126
- Logical drive 306, 315
- Long reference, link 233
- Lowercase letters 237
- Lowercase letters, preserving 222
- Making a directory 11, 94
- Making a subdirectory 94
- Manual
 - before you begin reading 1
 - how to use this manual x

- Notational conventions used in this manual ix
- other useful manuals x
- Map expanded memory pages command, debug
 - abbreviation 272
 - comments 272
 - purpose 272
 - syntax 272
- Map file
 - addresses 216
 - creating 209, 221
 - definition 216
 - format 211
 - including line numbers 221
 - including public symbols 216
 - segments 216
- Maximum memory allocation, controlling in link 224
- Media, disk
 - floppy 36, 119
 - hard 36, 119
- Mem command 92
 - defined viii
 - purpose 92
 - switch
 - /debug 92
 - /program 92
 - syntax 92
- Memory
 - Allocate expanded 270
 - allocating disk buffers 281
 - and disk buffers 281
 - Append command 186
 - computer 15
 - deallocate expanded 271
 - displaying 92
 - expanded 270, 271, 312
 - expanded memory 281, 311
 - expanded memory status 273
 - extended 312
 - extended memory 310
 - Map expanded memory pages 272
 - specifying with Select command 122
 - transient versus resident 47
 - Write command 205
- Memory, expanded
 - and MS-DOS 4.0 viii
- Memos directory 6
- Merging files, Edlin 204
- Message directory 335
- Messages
 - displaying on screen 155
 - printing on screen 155
- Misspelling, template 169
- Mistakes, correcting 165
- Mkdir command 94
 - purpose 94
 - shorthand notation 94
 - syntax 94
 - working directory 94
- Mnemonics, debug 242
- Mode
 - asynchronous communications 102
 - code page switching 100
 - console screen code pages 100
 - device code pages 100
 - inverse video 112
 - operation 95
 - parallel printer code pages 100
- Mode command
 - parallel printer mode
 - default settings 97
 - options 97
 - purpose 95
 - syntax 95
- Mode command viii
- Mode cp prepare command 330
- Modes
 - parallel printer modes 97
- Modes of operation 300
- Monochrome display adapter 100
- More command
 - purpose 105
 - syntax 105
- Move command, debug
 - abbreviation 256
 - comments 256
 - purpose 256
 - syntax 256
- Move command, Edlin
 - purpose 195
 - syntax 195
- Moving lines, Edlin 195
- MS-DOS
 - command processor 47, 68
 - exiting to from Edlin 198
 - file header 218
 - function keys 165
 - line editor, EDLIN 143
 - prepared code pages 327

- special editing keys 165
- special prompts 111
- Version 4.0
 - features of viii
- MS-DOS command
 - Append 29
 - Backup 120
 - Break 39
 - Chcp 40
 - Chdir 11, 42
 - Chdir (cd) 11
 - Chkdsk 3, 44
 - Command 47
 - Copy 16
 - Ctty 54
 - Date 55
 - Del 10, 16, 57
 - Dir 7, 10, 12, 16, 58
 - Diskcomp 60
 - Diskcopy 16, 63
 - diskcopy.exe. 16
 - Echo 155
 - Exe2bin 66
 - Exit 68
 - external 15, 16, 17
 - fc 71
 - Find 19
 - For 156
 - Format 3, 16
 - format.exe 16
 - Goto 158
 - Graphics 83
 - If 159
 - internal 15
 - Join 85
 - Keyb 87
 - Mkdir 11, 94
 - Mode 95
 - More 19
 - Nlsfunc 106
 - Path 17, 107
 - Pause 161
 - Print 108
 - Prompt 111
 - Recover 113
 - Rem 162
 - Ren 115
 - Replace 116
 - Restore 119
 - Rmdir 12, 121
 - Select 122
 - Set 124
 - Set, used with link 215
 - Share 126
 - Shift 149, 163
 - Sort 19, 128
 - Subst 130
 - Sys 131
 - Time 132
 - Tree 134
 - Type 16, 135
 - Ver 136
 - Verify 137
 - Vol 138
 - Xcopy 139
- MS-DOS commands
 - listing of 26
- MS-DOS prompt
 - allowed characters 111
 - command 111
 - default drive letter 275
 - default setting 111
 - special prompts 111
- MS-DOS version number, displaying 136
- Multilevel directory 3, 7
 - Mkdir command 94
 - removing a directory 121
 - using wildcards with 9
- Name command, debug
 - abbreviation 257
 - comments 257
 - purpose 257
 - syntax 257
- Named parameters 149
- National language support
 - code pages 327
 - MS-DOS 327
- Near segment-relative reference, link 233
- Near self-relative reference, link 233
- Network 22
 - Share command 126
- Network device
 - commanding to 22
- Network, symbol 22
- Nlsfunc command 330
 - config.sys processing 292
 - purpose 106

- syntax 106
- No paper error 380
- Non-DOS disk error 383
- Not parameter, If command 159
- Not ready error 384
- NUL data path 30, 107
- Numeric parameter 298
- Object file, link 208
- Omitting options, Edlin 194
- One-drive system 275
- Opcodes, debug 243
- Operating system
 - exiting to from Edlin 198
- Operating system, definition viii
- Operation modes 95
- Options, command 25
 - definition 24
- Order of segments, link 230
- Organizing files 4, 7
- Output 18
 - sending to lineprinter 171
 - stopping 171
- Output command, debug
 - abbreviation 259
 - comments 259
 - purpose 259
 - syntax 259
- Output, appending 18
- Output, redirecting 18
- Overlays, link 211, 227
- Packing executable files 220
- Page command, Edlin
 - purpose 197
 - syntax 197
- Paging through files, Edlin 197
- Parallel printer
 - code page switching 102
- Parameter
 - changing position of 163
 - column 298
 - condition 159
 - definition 149
 - line 298
 - not, If command 159
 - numeric 298
 - replaceable 149, 152
 - selective 298
 - shifting 163
- Parameter, named 150
- Parameter, replaceable 124, 150
- Parent directory
 - Chdir command 42
 - definition 8
 - parent directory 8
 - shorthand notation 8
- Parity 99
- Partition
 - DOS partition 315
 - creating 317
 - deleting 322
 - hard disk
 - definition 315
- Path
 - definition 24
 - NUL 107
 - specifying a working path 17
 - syntax 24
 - using with external commands 17
 - using with internal command 16
 - versus pathname 7
- Path command 17
 - purpose 107
 - syntax 107
- Pathname
 - as a command option 24
 - definition 7
 - maximum length 8
 - searching 8
 - syntax 7, 24
 - used with Chdir command 11
 - Using with external commands 17
 - using with internal commands 16
- Pathnames
 - Comp command 49
- Pause
 - displaying messages 161
- Pause command, batch
 - purpose 161
 - syntax 161
- Pause option 217
- Percent sign, used in a batch file 149
- Permission
 - read-only 34
 - write 34
- Pipe 18, 19
- Pipe symbol (|) 19

- Pipe, Sort command 129
- Pipeline, definition 20
- Pipes 128
- Piping and filtering
 - definition 19
- Piping and redirecting
 - definition 18
 - input 20
 - output 20
- Piping commands 19
- Placeholders 149
- Prepared code page 327
- Preserving lowercase characters, link 222
- Print command
 - purpose 108
 - switch
 - /b 108
 - /c 109
 - /d 108
 - /m 108
 - /p 109
 - /q 109
 - /s 108
 - /t 109
 - /u 108
 - syntax 108
- Print queue
 - cancel mode 109
 - deleting files from 109
 - print mode 109
- Print queue number of files allowed 109
- Printer 108
 - parallel
 - configuring 95, 97
 - modes 97
 - reconfiguring 95
 - redirecting output for parallel port to serial port 104
 - serial
 - configuring 99
 - serial communications port 99
- Printer options
 - Graphics command 83
- Printer port
 - specifying with Select command 123
- Printer type
 - specifying with Select command 123
- PRINTER.SYS 297
- Printer.sys
 - definition 309
 - device driver 286
 - installing 309
 - options 309
- Printing a Tree listing 134
- PRN, default printer 108
- Proceed command, debug
 - abbreviation 260
 - comments 260
 - purpose 260
 - syntax 260
- Processing, batch files 124, 143, 156
- Processing, interactive 156
- Processor, command 295
- Program
 - autoexec.bat 18
 - batch 143
 - COMMAND.COM 15
 - diskcopy.exe 16
 - format.exe 16
- Program, executable 210
- Programs directory 6, 8
- Prompt> 111
- Prompt
 - \$ 111
 - < 111
 - ASCII code (escape) 111
 - asterisk (*), Edlin 174
 - date 111
 - default drive 111
 - default drive letter 275
 - ENTER-LINEFEED 111
 - space 111
 - time 111, 132
 - version number 111
 - working directory, default drive 111
 - | 111
- Prompt command
 - purpose 111
 - syntax 111
- Prompt, MS-DOS 111
- Prompts
 - with Link 208
- Protecting files 2
- Public-symbol 216
- Public-symbol map 221
- Question mark (?)
 - wildcard character 9
- Question mark option, Edlin 202

- description 185
- Replace command 185
- Search command 185
- Queue, print 109
- Quit command
 - versus End command 198
- Quit command, debug
 - abbreviation 261
 - comments 261
 - purpose 261
 - syntax 261
- Quit command, Edlin
 - purpose 198
 - syntax 198
- Quitting
 - Edlin 175, 198
- Quitting without saving changes 198
- Quotation marks, use of 75
- QWERTY keyboard format 88
- RAMdisk 310
 - lost information in 311
- RAMDRIVE.SYS 297
- Ramdrive.sys
 - definition 310
 - device driver 286
 - installing 310
 - options 310
- RCP-Restore Cursor Position, ANSI escape sequence 300
- Read fault error 390
- Read-only
 - backup 176
- Read-only backup 191
- Read-only file, restoring 119
- Read-only files
 - replacing 116
- Read-only permission 34
- Recover command
 - purpose 113
 - syntax 113
- Recovering a file 113
- Redirecting and piping
 - input 20
 - output 20
- Redirection and piping symbols 145
- Redirection symbol
 - greater-than sign (>) 18
 - less-than sign (<) 19
 - two greater than signs (>>) 18
- Register command, debug
 - abbreviation 262
 - comments 262
 - purpose 262
 - syntax 262
- relative address (Fc) 71
- Rem command, batch
 - purpose 162
 - syntax 162
- Rem command, config.sys file
 - definition 278
- Rem, config.sys file
 - purpose 294
 - syntax 294
- Remark, batch file 162
- Removing
 - a directory 12, 121
 - a virtual drive 130
 - files 13, 57
- Removing groups from a program, link 226
- Ren command 175
 - purpose 115
 - syntax 115
 - using wildcards with 115
- Renaming a directory 13
- Renaming a file 115
- Replace command
 - exit codes 117
 - purpose 116
 - syntax 116
- Replace command, Edlin
 - purpose 199
 - syntax 199
- Replace mode, Edlin 180
- Replaceable parameters 124, 149, 150, 152
- Replacing a file 116, 118
- Replacing text, Edlin 199
- Responding to errors
 - Ignore 335
- Response file, link 213
- Restore command
 - country code 284
 - purpose 119
 - switch
 - /a 119
 - /b 119
 - /e 119
 - /L 119

- /m 119
- /n 119
- /p 119
- /s 119
- syntax 119
- Restoring
 - a file 119
 - a subdirectory 119
- Reverse sorting (Z to A) 128
- Reverse Video function, ANSI 300
- Revising text, Edlin 173
- RM-Reset Mode, ANSI escape sequence 301
- Rmdir command
 - hidden files 121
 - purpose 121
 - shorthand notation 13
 - syntax 121
- Root directory 8
 - autoexec.bat 145
 - config.sys file 277
 - definition 3, 43
 - joining with 86
 - making subdirectories 94
 - maximum capacity 4
 - shorthand notation 7
 - subdirectories of 6
- Running a batch file 151
- Sample config.sys file 278
- Saving a file, Edlin 175
- SCP-Save Cursor Position, ANSI escape sequence 299
- Screen
 - changing 298
 - code page switching 100
 - erasing 300
 - graphics 300
 - width, changing 301
- Scrolling
 - directory display 58
- Search command, debug
 - abbreviation 265
 - comments 265
 - purpose 265
 - syntax 265
- Search command, Edlin
 - purpose 202
 - syntax 202
- Search path 107
- Search paths
 - definition 215
 - Search paths, link 215
 - Searching a directory, Path 107
 - Searching for a file 5
 - Searching for text 19
 - Searching for text, Edlin 202
- Sector 113
- Sector not found error 393
- Seek error 393
- Segment alignment, link 230
- Segment number, setting maximum 228
- Segment order
 - MS-DOS convention 228
- Select command
 - MS-DOS 4.0 enhancements to viii
 - purpose 122
 - syntax 122
- Selective parameter 298
- Semicolon, used for options 25
- Separating files 3
- Separator
 - equal sign 158
 - semicolon 158
- Separators
 - decimal separator 278
 - definition 25
 - Time 132
- Separators in comments
 - comma 162
 - space 162
 - tab 162
- Sequences of commands 143
- Serial communications port
 - configuring 99
- Session Manager
 - starting from a subshell 295
- Set Active Partition menu 321
- Set command
 - purpose 124
 - syntax 124
- Set command, used with link 215
- Setting a working path 18
- SGR-Set Graphics Rendition, ANSI escape sequence 300
- Share command
 - config.sys processing 292
 - purpose 126
 - syntax 126
- Sharing a file 126

- Sharing violation 393
- Shell
 - definition 294
 - starting 294
 - starting Session Manager 295
- Shell command
 - use of with Command command 48
 - using switches 295
- Shell command, config.sys file
 - comments 295
 - definition 278
 - purpose 294
 - syntax 294
- Shift command 149
- Shift command, batch
 - purpose 163
 - syntax 163
- Shifting parameters 163
- Short reference, link 233
- Shortcuts
 - asterisk (*) 9
- Chdir command (cd) 12, 42
 - directory
 - “.” 121
 - “..” 121
- Mkdir command (md) 94
- Mode command
 - code page modes 103
- parent directory 8
- question mark (?) 9
- Rmdir command (rd) 13, 121
 - wildcard characters 9
 - working directory 8
- Sign, greater-than (>) 18
- Single-drive system 275
- Skipping over template characters 170
- SM-Set Mode, ANSI escape sequence 301
- Smartdrive.sys
 - device driver 286
- Smartdrv.sys
 - definition 312
 - installing 312
 - options 312
- Sort command 19
 - filter 129
 - pipe 129
 - purpose 128
 - switch
 - /+n 128
 - /r 128
 - syntax 128
- Sorter.bat file 149
- sorter.bat file 151
- Sorting
 - a directory 20
 - a file 19
 - alphabetically 151
 - files 128
 - in reverse order (Z to A) 128
- Source drive 24
- Space prompt 111
- Spaces and tabs, used for options 25
- Special editing key
 - DEL 167
 - ESC 167
 - F1 166
 - F2 166
 - F3 166
 - F4 167
 - F5 167
 - F6 167
 - INS 167
- Special editing keys
 - description 176
 - listing of 182
- Special editing keys, using with Edlin 173
- Specifying a working path 17
- Stack size
 - controlling with link 223
- Stacks command, config.sys file
 - definition 278
 - purpose 296
 - syntax 296
- Standard 128
- Starting
 - Edlin 174
- STDIN 128
- Stopbit 99
- Stopping a command 39, 171
- Stopping Edlin 175
- String
 - definition 25
 - environment 124, 145
 - finding 75
 - invalid separators
 - comma 159
 - equal sign 159
 - semicolon 159

- space 159
- replacing, Edlin 199
- searching, Edlin 202
- String, debug 240
- Structure
 - directory 5
 - file 5
 - hierarchical 5
 - multilevel 5
- Subdirectories
 - finding subdirectory names
 - Tree 134
 - naming 12
- Subdirectory 4, 6, 12
 - making 94
 - maximum capacity 4
- Subdirectory, restoring 119
- Subfunction 298
- Subst command
 - purpose 130
 - syntax 130
- Switch
 - definition 24
 - syntax 24
- Switches
 - /p switch 279
 - table, disk type and valid switches 80
- Switches, driver.sys
 - /c 306
 - /d 306
 - /f 306
 - /h 306
 - /s 306
 - /t 306
- Switches, drivparm
 - /c switch 287
 - /d switch 287
 - f switch 287
 - h switch 287
 - i switch 287
 - n switch 287
 - s switch 287
 - t switch 287
- Symbol
 - greater-than sign (>) 18
 - redirection 145
- Symbol-map file
 - format 216
- Symdeb 220, 221
- Synonym, Del, Erase 57
- Syntax
 - command 24
 - path 24
 - pathname 7, 24
 - switch 24
- Sys command
 - purpose 131
 - syntax 131
- System files, transferring 131
- System prompt 275
- System, file 4
- Target drive 24
- Template
 - advantages of 165
 - automatic overwriting 169
 - changing a template 168
 - creating a template 168
 - dealing with misspellings 169
 - deleting contents of 180
 - relation to command line 166
 - using the BKSP key with 181
 - using the DEL key with 178
 - using the ESC key with 179
 - using the F1 key with 177
 - using the F2 key with 177
 - using the F4 key with 179
 - using the F5 key with 180
 - using the INS key with 180
- Template character
 - copying 170
 - deleting 170
 - inserting 170
 - skipping over 170
- Temporary file
 - deleting 152
 - purpose 152
 - used in batch processing 149
 - used with the More command 105
- Temporary file, link 217
- Terminating a batch program 161
- Text editor, Edlin 11
- Text file, creating with Edlin 173
- Text option, Edlin
 - description 185
 - Replace command 185
 - Search command 185
- Text, moving, Edlin 195

- Text, searching for 19
- Time
 - changing current 132
 - Country command 278, 283
 - displaying current 132
- Time command
 - country code 284
 - purpose 132
 - syntax 132
- Time format 122
 - table, relation to different countries 284
- Time prompt 111, 132
- Time-out loop 97
- Trace command, debug
 - abbreviation 266
 - comments 266
 - purpose 266
 - syntax 266
- Transfer command, Edlin
 - purpose 204
 - syntax 204
- Transferring system files 131
- Transferring text, Edlin 204
- Transmission rate 99
- Tree command
 - purpose 134
 - syntax 134
- Tree command viii
- Two-letter code, Keyb 87
- Type command 16
 - purpose 135
 - syntax 135
- Typical config.sys file 278
- Unassemble command, debug
 - abbreviation 267
 - comments 267
 - purpose 267
 - syntax 267
- Underline, cursor 168
- Updating a file 116
 - .bak extension file 175
- Updating the system 131
- Uppercase letter, use of 183, 194
- Uppercase letters 222, 237
- User directory 6
- Using code pages 327
- Using control characters 170, 183
- Using Edlin command options 184
- Variable, replaceable 124, 150
- Ver command
 - purpose 136
 - syntax 136
- Verify command
 - purpose 137
 - syntax 137
- Version number prompt 111
- Version number, displaying 136
- Virtual disk 310
- Virtual drive
 - creating 130
 - definition 130
 - removing 130
- Vm.tmp file, link 217
- Vol command
 - purpose 138
 - syntax 138
- Volume label
 - allowed characters 91
 - creating 90
 - displaying 138
 - Label command 90
 - length 90
- Volume serial number 90
- Wide display 58
- Wildcard abbreviation (*.*) 10
- Wildcard character
 - asterisk (*) 9
 - destructive uses of 10
 - function of 9
 - question mark (?) 9
 - shortcuts 9
 - special uses 10
 - using the asterisk (*) 9
 - using the question mark (?) 9
 - using with Del 13
- Wildcards
 - use of with Comp command 49
- Working directory 6
 - definition 5
 - displaying 11, 42
 - external command directory 17
 - Mkdir command 94
 - program directory 17
 - prompt, default drive 111
 - searching for data files in 30
 - shorthand notation 8

- using filenames 8
- using pathnames 8
- Working path 17
- Write command, debug
 - abbreviation 269
 - comments 269
 - purpose 269
 - syntax 269
- Write command, Edlin
 - purpose 205
 - syntax 205
- Write fault error 403
- Write protect error 404
- Write-protect notch 2, 404
- Write-protect tab 404
- Write-protected disk 105
- Write-protection 2
- Writing edited lines to disk 186
- Writing to
 - a disk 137
 - a disk, Edlin 205
- Xcopy command
 - purpose 139
 - switch
 - /a 139
 - /d 139
 - /e 139
 - /m 139
 - /p 140
 - /s 140
 - /v 140
 - /w 140
 - syntax 139
 - versus Diskcopy command 140
- *.bak extension
 - updating a .bak file 175
- *MS-DOS
 - argument 25
 - command options 25
 - control characters 170, 183
 - function keys 176
 - special editing keys 176
 - switches 24
- *MS-DOS command
 - Assign 32
 - Attrib 34
 - Backup 36
 - Cls 46
 - Comp 49
 - Copy 51
 - Fastopen 69
 - Fdisk 74
 - Find 75
 - Format 77
 - Graftabl 81
 - Label 90
 - Mem 92
 - More 105
 - options 24
 - syntax 24
 - *MS-DOS partition
 - creating or deleting 74
 - FCBs (File control blocks)
 - FCBS command 278
 - swapping
 - defined 312
 - "Delete 322
 - "DOS 320, 322
 - "Extended 322
 - "Fdisk 317
 - "Hard 317
 - "Logical 320, 322
 - BKSP key 176
 - DEL key 166, 176
 - ESC key 166, 176
 - F1 key 166, 176
 - F2 key 166, 176
 - F3 key 166, 176
 - F4 key 166, 176
 - F5 key 166, 176
 - F6 key 166
 - INS key 166, 176
 - command
 - Call 154
 - ! prompt 111

—

1

—

